TECHNICAL REPORT RE-83-6

COMPARISON OF ARITHMETIC REQUIREMENTS FOR THE PFA, WFTA, SWIFT, MFFT, FFT AND DFT ALGORITHMS

Robert C. Hicks
Advanced Sensors Directorate
US Army Missile Laboratory

November 1982

U.S. ARMY MISSILE COMMAND
Redstone Arsenal, Alabama 35898

Approved for public release; distribution unlimited.

DISPOSITION INSTRUCTIONS

DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED. DO NOT RETURN IT TO THE ORIGINATOR.

DISCLAIMER

THE FINDINGS IN THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

TRADE NAMES

USE OF TRADE NAMES OR MANUFACTURERS IN THIS REPORT DOES NOT CONSTITUTE AN OFFICIAL INDORSEMENT OR APPROVAL OF THE USE OF SUCH COMMERCIAL HARDWARE OR SOFTWARE.

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
TR-RE-83-6

2. GOVT ACCESSION NO.
AD-A133 087

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
Comparison of Arithmetic Requirements for the PFA, WFTA, SWIFT, MFFT, FFT and DFT Algorithms

5. TYPE OF REPORT & PERIOD COVERED
Technical Report

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Robert C. Hicks

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Commander, US Army Missile Command
ATTN: DRSMI-RER
Redstone Arsenal, AL 35898

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
Commander, US Army Missile Command
ATTN: DRSMI-RPT
Redstone Arsenal, AL 35898

12. REPORT DATE
November 1982

13. NUMBER OF PAGES

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Discrete Fourier Transform
Fast Fourier Transform
Prime Factor Algorithm
Winograd Fourier Transform Algorithm
Mixed Radix Fast Fourier Transform
SWIFT
Arithmetic Requirements
Digital Signal Processing

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This discrete Fourier transform (DFT) is a powerful reversible mapping transform for discrete data sequences with mathematical properties analogous to those of the Fourier transform. The DFT can be used for spectral analysis of time series, fast correlation of sequences, fast convolution of sequences for the purpose of digital filtering, and for radar digital beamforming. The ever increasing importance of the DFT algorithm has led to the development of several more efficient algorithms requiring far less arithmetic computations
(Continued)

20. (Continued)

than the DFT. This report examines the multidimentional DFT decomposition theory central to many of these algorithms and gives a brief introduction to the radix-2 fast Fourier transform (FFT), radix-4 FFT, mixed radix fast Fourier transform (MFFT), prime factor algorithm (PFA), Winograd Fourier transform (WFTA), and SWIFT algorithms. In addition, the arithmetic complexity of these algorithms is compared for various one and two-dimensional transform sizes. Included in the comparison are the number of real additions, real multiplications, total real operations, total equivalent real multiplications, and integrated circuit chips required for each algorithm.

CONTENTS

I. INTRODUCTION

This report compares the arithmetic requirements of several efficient algorithms which compute the Discrete Fourier Transform (DFT). The DFT is a powerful, reversible mapping transform for discrete data sequences with mathematical properties analogous to those of the Fourier transform. The definitions of the DFT and the inverse DFT can be written in the form

$$A(k) = \sum_{n=0}^{N-1} x(n)\exp(-j2\pi nk/N) \tag{1}$$

$$x(n) = \sum_{k=0}^{N-1} A(k)\exp(j2\pi nk/N) \tag{2}$$

for k=0,1,..., N-1; n=0,1,..., N-1. The N-point data sequences x(n) and A(k) are generally complex and are often used to represent time and frequency series, respectively.

In 1965, Cooley and Tukey began a revolution in the field of signal processing when they introduced the Fast Fourier Transform (FFT) as an algorithm for efficiently computing the DFT [1]. The FFT reduced the number of computations requi-ed to compute the DFT from a number proportional to N^2, to one proportional to $N\log_2 N$. This reduction of comp''tations spurred widespread application of the DFT to many problems in diverse fields. In addition to spectral analysis of time series, the FFT has been used for fast correlation of sequences, fast convolution of sequences for the purpose of digital filtering, and radar Digital Beam Forming (DBF). In DBF applications, the output of each element of a receive-only array antenna is independently converted into complex baseband samples. A DFT is then used to transform the data into a simultaneous set of receive beams uniformly distributed in space [2].

The ever increasing importance of the DFT algorithm has led to the development of many new efficient algorithms requiring far less computations than the FFT. This report examines the multidimensional DFT decomposition theory central to many of these algorithms, and gives a brief introduction to the radix-2 FFT, radix-4 FFT, mixed radix fast Fourier transform (MFFT), prime factor algorithm (PFA), Winograd Fourier transform algorithm (WFTA), and SWIFT algorithms. In addition, the arithmetic complexity of these algorithms is compared for various one and two-dimensional transform sizes. Included in the comparison are the number of real additions, real multiplications, total real operations, total equivalent real multiplications, and integrated circuit chips required for each algorithm.

II. MULTIDIMENSIONAL DFT THEORY

All of the efficient DFT algorithms examined in this report are based on Good's standard multidimensional DFT decomposition technique [3-4]. This technique decomposes a large one-dimensional DFT into a sequence of smaller DFTs which are combined with twiddle factors (i.e., complex weights or multiplications). The number of multiplications and additions required to compute a DFT is greatly reduced by computing its decomposed small point DFT transforms, even though the twiddle factors increase the computational load.

However, the multidimensional decomposition is only applicable to the DFTs of length N, where N is factorable into integer values (i.e., $N = N_1*N_2*....,N_r$). In order to circumvent this requirement, DFTs can be appended with zeros to give a length that is factorable.

The basic mechanism of the multidimensional decomposition is transforming the one-dimensional data sequence of length $N = N_1*N_2$ into a two-dimensional rectangular array of N_1 rows and N_2 columns. The N-point DFT can then be computed by performing N_2-point DFTs on all the rows, and performing N_1-point DFTs on all the columns, and in some cases, multiplying the intermediate results by complex twiddle factors. If desired, the N_1 and N_2-point DFTs can be decomposed if they are factorable. This process can be applied repeatedly to the one-dimensional DFTs until the original N-point DFT has been completely decomposed into all of its integer factors.

A unique or one-to-one mapping function is needed to map the one-dimensional arrays A(k) and x(n) of the DFT expression

$$A(k) = \sum_n x(n)\, W_N^{nk} \qquad (3)$$

into the two-dimensional arrays $\hat{A}(k_1,k_2)$ and $\hat{x}(n_1,n_2)$ of the two-dimensional function [5]

$$\hat{A}(k_1,k_2) = \sum_{n_1} \sum_{n_2} \hat{x}(n_1,n_2)\, W_N^{nk} \qquad (4)$$

where k_1, $n_1 = 0,1,...,$ $N_1 - 1$; k_2, $n_2 = 0,1,...,$ $N_2 - 1$; and $W_N^{nk} = \exp(-j2\pi nk/N)$. Although many different mapping functions exist, the mapping function fundamental to most fast DFT algorithms is

$$\left\{\begin{array}{l} n = (L_1n_1 + L_2n_2)\mathrm{mod}\ N \\ k = (L_3k_1 + L_4k_2)\mathrm{mod}\ N \end{array}\right\} . \qquad (5)$$

A simple mapping of this form is

$$\left\{\begin{array}{l} n = (n_1 + N_1n_2)\mathrm{mod}\ N \\ k = (N_2k_1 + k_2)\mathrm{mod}\ N \end{array}\right\} . \qquad (6)$$

For example, this mapping can be used to decompose the vectors A(k) and x(n) of an eight-point DFT into two-dimensional functions with N_1 rows and N_2 columns. For the values $N_1 = 2$ and $N_2 = 4$, the mapping between x(n) and $\hat{x}(n_1,n_2)$ is

$$\hat{x}(n_1,n_2) = x(n_1 + 2n_2)\mathrm{mod}\ 8 , \qquad (7)$$

as shown below:

$$\begin{bmatrix} \hat{x}(0,0) = x(0) & \hat{x}(0,1) = x(2) & \hat{x}(0,2) = x(4) & \hat{x}(0,3) = x(6) \\ \hat{x}(1,0) = x(1) & \hat{x}(1,1) = x(3) & \hat{x}(1,2) = x(5) & \hat{x}(1,3) = x(7) \end{bmatrix} . \qquad (8)$$

Note that each position in the above 2x4 matrix is assigned a unique value from the x(n) vector. The mapping for the output values is

$$\hat{A}(k_1,k_2) = A(4k_1 + k_2)\text{mod } 8 \tag{9}$$

as shown below:

$$\hat{A}(0,0) = A(0) \quad \hat{A}(0,1) = A(1) \quad \hat{A}(0,2) = A(2) \quad \hat{A}(0,3) = A(3)$$
$$\hat{A}(1,0) = A(4) \quad \hat{A}(1,1) = A(5) \quad \hat{A}(1,2) = A(6) \quad \hat{A}(1,3) = A(7) \; . \tag{10}$$

The mapping of (5) can be substituted into Equation (4) giving

$$\hat{A}(k_1,k_2) = \sum_{n_1} \sum_{n_2} \hat{x}(n_1,n_2) W_N^{(L_1n_1 + L_2n_2)(L_3k_1 + L_4k_2)}$$

or

$$\hat{A}(k_1,k_2) = \sum_{n_1} \sum_{n_2} \hat{x}(n_1,n_2) W_N^{L_2L_4n_2k_2} W_N^{L_1L_4n_1k_2} W_N^{L_1L_3n_1k_1} W_N^{L_2L_3n_2k_1} \; . \tag{11}$$

where

$$\hat{A}(k_1,k_2) \equiv A(L_3k_1 + L_4k_2)\text{mod } N$$

$$\hat{x}(n_1,n_2) \equiv x(L_1n_1 + L_2n_2)\text{mod } N$$

L_1, L_2, L_3, and L_4 can be selected using the results of a theorem from number theory to insure a unique mapping. Case A of the theorem applies when the factors N_1 and N_2 are mutually prime, that is 1 is the largest common integer factor. Case B applies when N_1 and N_2 are not mutually prime, that is N_1 and N_2 have a common integer factor, λ , which is greater than 1. The notation used in the theorem to represent these two cases is

CASE A: $(N_1,N_2) = 1$

CASE B: $(N_1,N_2) = \lambda$, (12)

where the operator (N_1, N_2) is defined as the greatest common integer factor of N_1 and N_2. The theorem can be written in terms of n or k of Equation (5) as they are of the same form. For simplicity, however, the theorem will be expressed for both the n and k mapping.

Theorem: The necessary and sufficient conditions for the mapping of Expression (5) to be unique are:

CASE A:

1) $L_1 = \alpha N_2$ and $L_2 \neq \beta N_1$ and $(\alpha,N_1) = (L_2,N_2) = 1$

$$L_3 = \gamma N_2 \text{ and } L_4 \neq \delta N_1 \text{ and } (\gamma,N_1) = (L_4,N_2) = 1 \tag{13}$$

2) $L_1 \neq \alpha N_2$ and $L_2 = \beta N_1$ and $(L_1, N_1) = (\beta, N_2) = 1$

$$L_3 \neq \gamma N_2 \text{ and } L_4 = \delta N_1 \text{ and } (L_3, N_1) = (\delta, N_2) = 1 \qquad (14)$$

3) $L_1 = \alpha N_2$ and $L_2 = \beta N_1$ and $(\alpha, N_1) = (\beta, N_2) = 1$

$$L_3 = \gamma N_2 \text{ and } L_4 = \delta N_1 \text{ and } (\gamma, N_1) = (\delta, N_2) = 1 \qquad (15)$$

CASE B:

1) $L_1 = \alpha N_2$ and $L_2 \neq \beta N_1$ and $(\alpha, N_1) = (L_2, N_2) = 1$

$$L_3 = \gamma N_2 \text{ and } L_4 \neq \delta N_1 \text{ and } (\gamma, N_1) = (L_4, N_2) = 1 \qquad (16)$$

2) $L_1 \neq \alpha N_2$ and $L_2 = \beta N_1$ and $(L_1, N_1) = (\beta, N_2) = 1$

$$L_3 \neq \gamma N_2 \text{ and } L_4 = \delta N_1 \text{ and } (L_3, N_1) = (\delta, N_2) = 1 \qquad (17)$$

The variables L_1, L_2, α, and β of this theorem will be used for the mapping of n and the variables L_3, L_4, γ, and δ will be used for the mapping of k. All of these variables are non-zero positive integers.

CASE B of the theorem will be considered first as it is the basis of the Decimation-In-Time (DIT) and Decimation-In-Frequency (DIF) algorithms. These algorithms are used to implement the familiar radix-2 and radix-4 FFT algorithms.

The DIT algorithm is derived by using Equation (17) for the mapping of n and Equation (16) for the mapping of k. Combining these expressions with that of (5) gives the mapping

$$\left\{\begin{array}{l} n = (L_1 n_1 + \beta N_1 n_2) \bmod N \\ k = (\gamma N_2 k_1 + L_4 k_2) \bmod N \end{array}\right\} , \qquad (18)$$

where $L_1 \neq \alpha N_2$ and $L_4 \neq \delta N_1$.

Substituting this into Equation (11) gives

$$\hat{A}(k_1 k_2) = \sum_{n_1} \sum_{n_2} \hat{x}(n_1, n_2) W_{N_2}^{\beta L_4 n_2 k_2} W_N^{L_1 L_4 n_1 k_2} W_{N_1}^{L_1 \gamma n_1 k_1} . \qquad (19)$$

Note that the last term of Equation (11) is eliminated as

$$\begin{aligned} W_N^{\beta\gamma n_2 k_1 (N_1 N_2)} &= \exp(-j2\pi\beta\gamma n_2 k_1 N/N) \\ &= (\exp(-j2\pi))^{\beta\gamma n_2 k_1} = 1 . \qquad (20) \end{aligned}$$

Choosing the values $L_1 = L_4 = \beta = \gamma = 1$ satisfies the theorem and when substituted into Equation (19) gives

$$\hat{A}(k_1, k_2) = \sum_{n_1} \sum_{n_2} \hat{x}(n_1, n_2) W_{N_2}^{n_2 k_2} W_N^{n_1 k_2} W_{N_1}^{n_1 k_1} , \qquad (21)$$

where

$$\hat{A}(k_1,k_2) = A(N_2k_1 + k_2)\text{mod } N$$

$$\hat{x}(n_1,n_2) = x(n_1 + N_1n_2)\text{mod } N .$$

The $W_N^{n_1k_2}$ term is the twiddle factor.

A brute force computation of Equation (21) would require N complex multiplications and N-1 complex additions for each value of the $\hat{A}(k_1,k_2)$ array, assuming prior combination of the three complex exponential terms. This would require N^2 complex multiplications and N(N-1) complex additions to compute the DFT. Fortunately, the number of operations required can be reduced by using one-dimensional DFTs on the rows and columns as suggested by the following nesting of Equation (21)

$$\hat{A}(k_1,k_2) = \sum_{n_1=0}^{N_1-1} W_{N_1}^{n_1k_1}\left[W_N^{n_1k_2}\left[\sum_{n_2=0}^{N_2-1} \hat{x}(n_1,n_2)W_{N_2}^{n_2k_2}\right]\right] . \qquad (22)$$

The innermost bracket is a function of n_1 and k_2

$$q(n_1,k_2) = \sum_{n_2} \hat{x}(n_1,n_2)W_{N_2}^{n_2k_2} \qquad (23)$$

where $k_2 = 0,1,\ldots,N_2-1$ and n_1 is fixed by the value of the outermost summation symbol. This is obviously an N_2-point one-dimensional DFT on the n_1th row of data. As indicated by the next level of brackets, each of the N $q(n_1,k_2)$ values is multiplied by its complex twiddle factor. The results of the two innermost brackets is still a function of n_1 and k_2

$$h(n_1,k_2) = q(n_1,k_2)W_N^{n_1k_2} . \qquad (24)$$

Combining Equations (22), (23), and (24) gives

$$\hat{A}(k_1,k_2) = \sum_{n_1=0}^{N_1-1} h(n_1,k_2)W_{N_1}^{n_1k_1} , \qquad (25)$$

where $k_1, = 0,1,\ldots,N_1-1$ and k_2 is a fixed value for each column of data. This is obviously an N_1-point one-dimensional DFT on the K_2th column. Thus, using the nesting of Equation (22), the N-point DFT is calculated by: (1) calculating an N_2-point one-dimensional DFT on the data of each of the N_1 rows; (2) multiplying each intermediate transformed data point by a complex twiddle factor; and (3) performing an N_1-point one-dimensional DFT on the twiddled data of each of the N_2 columns. The required real multiplications and additions for this process can be expressed

$$NRMULT = N_1\mu_2 + N_2\mu_1 + 4N \qquad (26)$$

$$NRADDS = N_1\alpha_2 + N_2\alpha_1 + 2N , \qquad (27)$$

where

$\mu_i \equiv$ number of real multiplications in the N_i-point DFT

$\alpha_i \equiv$ number of real additions in the N_i-point DFT.

This method is generally more efficient than the brute force computation of Equation (21). Greater efficiency results if the N_1 and/or N_2 one-dimensional DFT(s) of the above process can be decomposed into still smaller factors.

The DIT Equation (21) can also be nested as

$$\hat{A}(k_1,k_2) = \sum_{n_1} W_{N_1}{}^{n_1k_1}\left[\sum_{n_2} W_{N_2}{}^{n_2k_2}\left[\hat{x}(n_1,n_2)W_N{}^{n_1k_2}\right]\right] . \tag{28}$$

The computation suggested by this nesting is very similar to that of Equation (22) as only the first two computation steps are reversed. For this nesting the N-point DFT is calculated by: (1) multiplying each data point by the appropriate complex twiddle factor; (2) calculating an N_2-point DFT of each row of the intermediate data; and (3) performing an N_1-point DFT of each column of the data calculated in step 2. The arithmetic requirements for computing Equation (28) are obviously the same as Equation (22).

A final way the DIT Equation (21) can be nested is

$$\hat{A}(k_1,k_2) = \sum_{n_2} W_{N_2}{}^{n_2k_2}\left[\sum_{n_1} W_{N_1}{}^{n_1k_1}\left[\hat{x}(n_1,n_2)W_N{}^{n_1k_2}\right]\right] . \tag{29}$$

For this reverse nesting, the N-point DFT is calculated by: (1) multiplying each data point by the appropriate twiddle factor; (2) calculating an N_1-point DFT of each column of the twiddled data; and (3) calculating an N_2-point DFT of each row. This also has the same arithmetic requirements as the other DIT nestings of Equations (22) and (28).

The DIF algorithm is obtained by using Equation (16) for the mapping of n and Equation (17) for the mapping of k. Combining these expressions with that of (5) gives the mapping

$$\left\{\begin{array}{l} n = \alpha N_2 n_1 + L_2 n_2 \bmod N \\ k = L_3 k_1 + \delta N_1 k_2 \bmod N \end{array}\right\} \tag{30}$$

where $L_2 \neq \beta N_1$ and $L_3 \neq \gamma N_2$.

Substituting this into Equation (11) gives

$$\hat{A}(k_1,k_2) = \sum_{n_1}\sum_{n_2} \hat{x}(n_1,n_2)W_{N_2}{}^{\delta L_2 n_2 k_2}W_{N_1}{}^{\alpha L_3 n_1 k_1}W_N{}^{L_2 L_3 n_2 k_1} . \tag{31}$$

Note that this combination of CASE B eliminates the second term of Equation (11) as

$$W_N{}^{\alpha\delta n_1 k_2(N_2 N_1)} = \exp(-j2\pi\alpha\delta n_1 k_2 N/N) = 1 . \tag{32}$$

Choosing the values $L_2 = L_3 = \alpha = \delta = 1$ satisfies the theorem and, when substituted into Equation (31), gives

$$\hat{A}(k_1,k_2) = \sum_{n_1}\sum_{n_2} \hat{x}(n_1,n_2)W_{N_2}^{n_2k_2}W_{N_1}^{n_1k_1}W_N^{n_2k_1} \ , \tag{33}$$

where

$$\hat{A}(k_1,k_2) = A(k_1 + N_1k_2)\bmod N$$

$$\hat{x}(n_1,n_2) = x(N_2n_1 + n_2)\bmod N \ .$$

The $W_N^{n_2k_1}$ term is the twiddle factor. Like the DIT algorithm, the DIF algorithm requires on the order of N^2 complex operations until nested according to one of the following three expressions:

$$\hat{A}(k_1,k_2) = \sum_{n_1=0}^{N_1-1} W_{N_1}^{n_1k_1}\left[\sum_{n_2=0}^{N_2-1} W_{N_2}^{n_2k_2}\left[\hat{x}(n_1,n_2)W_N^{n_2k_1}\right]\right] \tag{34}$$

$$\hat{A}(k_1,k_2) = \sum_{n_2=0}^{N_2-1} W_{N_2}^{n_2k_2}\left[W_N^{n_2k_1}\left[\sum_{n_1=0}^{N_1-1} \hat{x}(n_1,n_2)W_{N_1}^{n_1k_1}\right]\right] \tag{35}$$

$$\hat{A}(k_1,k_2) = \sum_{n_2=0}^{N_2-1} W_{N_2}^{n_2k_2}\left[\sum_{n_1=0}^{N_1-1} W_{N_1}^{n_1k_1}\left[\hat{x}(n_1,n_2)W_N^{n_2k_1}\right]\right] \ . \tag{36}$$

Equations (34) and (36) are calculated like Equations (28) and (29), respectively. Only the twiddle factors and the mapping are different. Equation (35) is calculated by (1) calculating an N_1-point DFT of each column, (2) twiddling the results, and (3) calculating an N_2-point DFT on each row of the twiddled results. All three nested expressions of the DIF algorithms require the same amount of computations as the nested DIT algorithms.

The two other possible combinations of CASE B of the theorem involve using Equations (16) or (17) for both the n and k mapping. However, neither of these maps allow the elimination of a complex exponential term of Equation (11). This prevents the efficient nesting of the two-dimensional function of Equation (11).

CASE A of the theorem is the basis of all DFT algorithms involving mutually prime factors, including the WFTA, PFA, and the SWIFT algorithms. Using Expression (15) of the theorem for n and k gives the mapping

$$\left\{\begin{array}{l} n = (\alpha N_2n_1 + \beta N_1n_2)\bmod N \\ k = (\gamma N_2k_1 + \delta N_1k_2)\bmod N \end{array}\right\} \ . \tag{37}$$

Substituting this into Equation (11) gives

$$\hat{A}(k_1,k_2) = \sum_{n_1}\sum_{n_2} \hat{x}(n_1,n_2)W_{N_2}^{\beta\delta N_1n_2k_2}W_{N_1}^{\alpha\gamma N_2n_1k_1} \ . \tag{38}$$

Note that both the second and fourth complex exponential term of Equation (11) are eliminated by this mapping. Good [3] suggested using the values

$$\left\{\begin{array}{l}\alpha = \beta = 1 \\ \delta = N_1{}^{-1}\text{mod } N_2 \\ \gamma = N_2{}^{-1}\text{mod } N_1\end{array}\right\} \qquad (39)$$

where δ and γ are multiplicative inverses of N_1 and N_2, respectively. The multiplicative inverse of a number N is defined as the unique integer, A, which belongs to the set (0,1,...,M-1) and satisfies

$$(A*N)\text{mod } M = 1 \ . \qquad (40)$$

For example, if $N_1 = 3$ and $N_2 = 7$ are used in Equations (37) and (39) then $\delta = 5$ and $\gamma = 1$, giving the mapping

$$\left\{\begin{array}{l}n = (7n_1 + 3n_2)\text{mod } 21 \\ k = (7k_1 + 15k_2)\text{mod } 21\end{array}\right\} \ . \qquad (41)$$

The multiplicative inverses of Equation (39) are guaranteed to exist because N_1 and N_2 have been restricted to being mutually prime for CASE A. Substituting the values of Equation (39) into Equation (38) gives

$$\hat{A}(k_1,k_2) = \sum_{n_1} \sum_{n_2} \hat{x}(n_1,n_2) W_{N_2}{}^{n_2k_2} W_{N_1}{}^{n_1k_1} \ , \qquad (42)$$

where

$$\hat{A}(k_1,k_2) = A((N_2{}^{-1}\text{mod} N_1)N_2k_1 + (N_1{}^{-1}\text{mod} N_2)N_1k_2)\text{mod } N$$

$$\hat{x}(n_1,n_2) = x(N_2n_1 + N_1n_2)\text{mod } N \ .$$

Because there is no twiddle factor, Equation (42) can be computed like a two-dimensional DFT. Thus, a DFT of length $N = N_1*N_2$ where $(N_1,N_2) = 1$ can be computed according to the two obvious nesting arrangements

$$\hat{A}(k_1,k_2) = \sum_{n_1=0}^{N_1-1} W_{N_1}{}^{n_1k_1} \left[\sum_{n_2=0}^{N_2-1} \hat{x}(n_1,n_2) W_{N_2}{}^{n_2k_2}\right] , \qquad (43)$$

$$\hat{A}(k_1,k_2) = \sum_{n_2=0}^{N_2-1} W_{N_2}{}^{n_2k_2} \left[\sum_{n_1=0}^{N_1-1} \hat{x}(n_1,n_2)\ W_{N_1}{}^{n_2k_1}\right] . \qquad (44)$$

The N-point DFT as nested in Equation (43) can be calculated by performing N_2-point DFTs on all N_1 rows of the data and performing N_1-point DFTs on all N_2 columns of the intermediate data resulting from step 1. The nesting of Equation (44) simply dictates calculating the column DFTs before calculating the row DFTs. The above method is referred to as the row/column technique. For the above two cases the computational requirement for calculating an $N = N_1*N_2$ point DFT where $(N_1,N_2) = 1$ is

$$\text{NRMULT} = N_1\mu_2 + N_2\mu_1 \tag{45}$$

$$\text{NRADDS} = N_1\alpha_2 + N_2\alpha_1 \ . \tag{46}$$

If N_2 can be factored further such that

$$N_2 = N_3 * N_4 \ , \tag{47}$$

where N_3 and N_4 are mutually prime, then the arithmetic requirements for computing the N_2-point DFT are

$$\mu_2 = N_3\mu_4 + N_4\mu_3 \tag{48}$$

$$\alpha_2 = N_3\alpha_4 + N_4\alpha_3 \ . \tag{49}$$

Thus, if N is factored such that

$$N = N_1 * N_3 * N_4 \ , \tag{50}$$

where all the factors are mutually prime, Equations (48) and (49) can be substituted into Equations (45) and (46) to give the requirements

$$\text{NRMULT} = N_1 N_2\mu_3 + N_3 N_1\mu_2 + N_2 N_3\mu_1 \tag{51}$$

$$\text{NRADDS} = N_1 N_2\alpha_3 + N_3 N_1\alpha_2 + N_2 N_3\alpha_1 \ . \tag{52}$$

In general, when N is factored into r mutually prime factors

$$N = N_1 * N_2 * \ldots . N_r \ , \tag{53}$$

the arithmetic requirements are simply

$$\text{NRMULT} = N \sum_{i=1}^{r} \frac{\mu_i}{N_i} \tag{54}$$

$$\text{NRADDS} = N \sum_{i=1}^{r} \frac{\alpha_i}{N_i} \tag{55}$$

III. EFFICIENT DFT ALGORITHMS

The radix-2 FFT is restricted to lengths N where N is a power of 2 (i.e., $N = 2^r$) [6]. The radix-2 algorithm is based on a complete decomposition of the N-point DFT into r 2-point DFTs. For N = 2 the DFT definition (see Equation (1)) simplifies to

$$A(0) = x(0) + x(1) \tag{56}$$

$$A(1) = x(0) - x(1) \ . \tag{57}$$

Thus, only two complex additions are required for each 2-point DFT. As shown in the last section, however, twiddle factors or complex multiplications are required between each 2-point DFT as the factors of N are not mutually prime.

The number of real multiplications and additions required for an N-point radix-2 FFT can be expressed as

$$\text{NRMULT} = 2N\log_2 N \qquad (58)$$

$$\text{NRADDS} = 3N\log_2 N \,. \qquad (59)$$

The radix-4 FFT is restricted to lengths N where N is a power of 4 (i.e., $N = 4^r$) [6]. The radix-4 algorithm only partially decomposes N into r 4-point DFTs. The 4-point DFT also requires no multiplications as shown in Appendix B. Like the radix-2 FFT, the radix-4 FFT requires complex multiplications because of twiddle factors. However, the radix-4 FFT requires 25% less multiplications than the radix-2 FFT as the former has fewer small point DFTs to connect with twiddle factors. An N-point radix-4 FFT requires

$$\text{NRMULT} = (3N/2)\log_2 N \qquad (60)$$

$$\text{NRADDS} = (11N/4)\log_2 N \,. \qquad (61)$$

The MFFT was published by Singleton in 1969 [7]. The MFFT can compute the DFT of any sequence length. N must be factored as

$$N = 2^r 3^s 4^t 5^u p_1{}^{m_1} p_2{}^{m_2} \cdots p_k{}^{m_k} \,, \qquad (62)$$

where the p_i's represent odd prime numbers. The arithmetic reqirements of the MFFT were determined [8] to be

$$\text{NRMULT} = 2rN + 4sN + 3tN + 32uN/5 +$$

$$\sum_{i=1}^{k} [2(p_i-1) + (m_i)N(p_i-1)^2/p_i + 4(m_i)N(p_i-1)/p_i] - 4(N-1) \qquad (63)$$

$$\text{NRADDS} = 3rN + 16sN/3 + 11tN/2 + 8uN +$$

$$\sum_{i=1}^{k} [(p_i-1) + 7N(m_i)(p_i-1)/p_i + (m_i)N(p_i-1)^2/p_i] - 2(N-1) \,. \qquad (64)$$

For the comparison purposes of this report, the arithmetic requirements of the MFFT were only calculated for the lengths suitable for the other efficient algorithms. The arithmetic requirements based on the restricted factorization

$$N = 2^r*3^s*4^t*5^u*7^w \,, \qquad (65)$$

can be expressed

$$\text{NRMULT} = N(2r + 4s + 3t + 32u/5 + 60w/7 - 4) + 12w + 4 \qquad (66)$$

$$\text{NRADDS} = N(3r + 16s/3 + 11t/2 + 8u + 78w/7 - 2) + 6w + 2 \,. \qquad (67)$$

The SWIFT algorithm is based on the standard multidimensional DFT decomposition which results when all the factors of N are mutually prime [9]. As shown by Equations (43) and (44) of the last section, no twiddle factors are required for this algorithm. Thus, as discussed in the last section, the arithmetic requirements of an N-point SWIFT algorithm with r mutually prime factors are

$$\text{NRMULT} = N \sum_{i=1}^{r} \frac{\mu_i}{N_i} \tag{68}$$

$$\text{NRADDS} = N \sum_{i=1}^{r} \frac{\alpha_i}{N_i} . \tag{69}$$

The SWIFT algorithm uses efficient small point DFT algorithms of lengths 2,3,4,5,6,7,8,9, and 16. Table 3-1 gives the number of non-trivial multiplications and additions required for each of these small point DFTs.

TABLE 3-1. SWIFT SHORT DFT REAL OPERATIONS REQUIREMENTS

N	μ_i	α_i
2	0	4
3	4	12
4	0	16
5	16	32
7	36	60
8	4	52
9	44	88
16	24	144

A listing of the algorithms is given in Appendix C. The different mutually prime combinations of these small point DFTs allow the SWIFT algorithm to compute DFTs of lengths N = 2 to N = 5040.

The PFA [10-11] is also based on the standard multidimensional DFT decomposition which results when the factors of N are mutually prime. Accordingly, the arithmetic requirements of an N-point PFA algorithm with r mutually prime factors are

$$\text{NRMULT} = N \sum_{i=1}^{r} \frac{\mu_i}{N_i} \tag{70}$$

$$\text{NRADDS} = N \sum_{i=1}^{r} \frac{\alpha_i}{N_i} . \tag{71}$$

The PFA also uses efficient small point DFTs of lengths 2,3,4,5,7,8,9, and 16. The number of non-trivial real multiplications and additions required for each of these small point DFTs is given in Table 3-2.

TABLE 3-2. PFA SHORT DFT REAL OPERATIONS REQUIREMENTS

N	μ_i	α_i
2	0	4
3	4	12
4	0	16
5	10	34
7	16	72
8	4	52
9	20	84
16	20	148

A listing of the algorithms is given in Appendix B.

The WFTA [12-17] was first published by Dr. Samuel Winograd in the mid-seventies. Like the SWIFT and PFA, the WFTA is based on a mutually prime factorization of N resulting in no twiddle factors. However, the WFTA offers an alternative to the row/column evaluations of Equation (42) used in the SWIFT and PFA. The WFTA uses the special structure of the WFTA short DFT transforms to nest all the multiplications inside of input and output additions. The number of real multiplications required of an N-point WFTA algorithm with r mutually prime factors is

$$\text{NRMULT} = 2\prod_{i=1}^{r}\delta_i - 2\prod_{i=1}^{r}\beta_i \ , \qquad (72)$$

where

$\delta_i \equiv$ the number of complex multiplications in the N_i-point DFT

$\beta_i \equiv$ the number of multiplications by "1" in the N_i-point DFT.

The number of real additions required [10] for two, three, and four factors is expressed in Equations (73), (74), and (75), respectively.

$$\text{NRADDS} = 2N_1\gamma_2 + 2\delta_2\gamma_1 \qquad (73)$$

$$\text{NRADDS} = 2N_1N_2\gamma_3 + 2\delta_3[N_1\gamma_2 + \delta_2\gamma_1] \qquad (74)$$

$$\text{NRADDS} = 2N_1N_2N_3\gamma_4 + 2\delta_4[N_1N_2\gamma_3 + \delta_3[N_1\gamma_2 + \delta_2\gamma_1]] \ , \qquad (75)$$

where $\gamma_i \equiv$ the number of complex additions in the N_i-point DFT. The WFTA also uses efficient small point DFTs of lengths 2,3,4,5,7,8,9, and 16. The total number of complex multiplications, the number of multiplications by "1," and the number of complex additions required for each of these small point DFTs is given in Table 3-3.

TABLE 3-3. WFTA SHORT DFT COMPLEX OPERATIONS REQUIREMENTS

N	δ_i	β_i	γ_i
2	2	2	2
3	3	1	6
4	4	3	8
5	6	1	17
7	9	1	36
8	8	4	26
9	11	1	44
16	18	5	74

The ordering of the factors of N can affect the number of real additions required by the WFTA. In this report the optimum ordering of the factors of N was always used to calculate WFTA real addition requirements. This optimum ordering is shown in Tables 4-7 and 4-8. A listing of the WFTA small point algorithms is given in Appendix A.

IV. COMPARISON OF ALGORITHM ARITHMETIC REQUIREMENTS

The arithmetic requirements of the various one-dimensional DFT algorithms are given in this chapter for lengths N = 2 to N = 5040. In addition, the requirements for various two-dimensional DFT algorithms are given for sizes ranging from 2x2 to 90x90. The one-dimensional requirements for the DFT, radix-2 FFT, radix-4 FFT, MFFT, SWIFT, WFTA, and PFA algorithms are compared in Tables 4-1 through 4-8. The two-dimensional requirements for the custom DFT, DFT, radix-2 FFT, MFFT, SWIFT, WFTA, and PFA algorithms are compared in Tables 4-9 through 4-12. In addition, Table 4-13 summarizes the chapter by listing the number of current and future chips required for various one and two-dimensional transforms. Tables 4-1 through 4-13 are located at the end of this chapter.

Tables 4-1 and 4-2 give the total number of real operations (i.e., the sum of the real multiplications and real additions) required for the one-dimensional DFT algorithms of N = 2 through N = 5040. Measured by the number of real operations, the DFT is by far the least efficient algorithm. For example, the DFT requires 54,600% of the operations required of the radix-2 FFT for N = 4096. A radix-4 FFT requires 85% of the real operations required by the radix-2 FFT. The MFFT requires fewer real operations than the DFT and the two FFTs. In addition, the MFFT can be used for every sequence length listed between N = 2 and N = 5040. However, the MFFT usually requires about 10% more operations than the PFA and WFTA. Generally, the PFA and WFTA are the most efficient algorithms for the lengths between N = 20 and N = 5040. The number of real operations required for the PFA, WFTA, and SWIFT algorithms are within 10% of the number required by the best algorithm for 100%, 96%, and 44% of the lengths in this range, respectively.

Generally, the multiplication operation requires more time and hardware resources than the addition operation. This is also true at the chip level where a multiplier requires approximately four times the silicon "real estate" of an adder. Accordingly, a weighted index of arithmetic complexity is particularly important if a custom chip can be designed to match the requirements of an algorithm. The weighted unit shown in Tables 4-3 and 4-4 is the Total

Equivalent Real Multiplications (TERM). The TERM unit is simply the total of the required real multiplications added to one-fourth the number of required real additions. As with the total number of real operations, the TERM count shows the DFT to be by far the least efficient algorithm. For example, the DFT requires from 700% to 62,000% of the TERM of comparable radix-2 FFTs. A radix-4 FFT requires about 80% of the TERM required by the radix-2 FFT. As before, the TERM index shows the MFFT and SWIFT algorithms to be more efficient than the DFT and FFT algorithms. However, for many of the lengths, the MFFT and SWIFT algorithms require up to 200% and 150% of the TERM required of the WFTA algorithm. For lengths between N = 20 and N = 5040, the WFTA is the most efficient algorithm for 93% of the lengths with the PFA being the most efficient algorithm for the other 7%. The required TERM for the WFTA, PFA, SWIFT, and MFFT algorithms are within 10% of the number required by the best algorithm for 100%, 58%, 2%, and 0% of the lengths in this range.

Tables 4-5 and 4-6 give the number of real multiplications required for one-dimensional DFT algorithms of lengths N = 2 through N = 5040. Once again, the DFT is by far the least efficient algorithm in terms of real multiplications. For example, the DFT requires 53,300% of the real multiplications needed for the radix-2 FFT for N = 4096. A radix-4 FFT requires 75% of the real multiplications required by the radix-2 FFT. The MFFT offers considerable savings in the number of multiplications required compared to the DFT and the FFT algorithms. However, the MFFT is never within 10% of the arithmetic requirement of the most efficient algorithms for lengths greater than N = 4. The WFTA is superior at minimizing the number of required multiplications. The WFTA is the most efficient algorithm in terms of real multiplications for 93% of the lengths between N = 20 and N = 5040. The PFA is the most efficient algorithm for the other 7% of the lengths. The percentages of the lengths in this range at which the MFFT, SWIFT, WFTA, and PFA algorithms are within 10% of the most efficient algorithms are 0%, 2%, 100%, and 9%, respectively.

Tables 4-7 and 4-8 give the number of real additions required for one-dimensional DFT algorithms of lengths N = 2 through N = 5040. In terms of real additions, the DFT is the least efficient algorithm followed in order of increasing efficiency by the radix-2 FFT and the radix-4 FFT. The radix-4 FFT requires 92% of the real additions required of the radix-2 FFT. The SWIFT algorithm is the most efficient, or as efficient, as any other algorithm for 98% of the lengths N = 20 to N = 5040. The percentages of the lengths in this range at which the SWIFT, PFA, WFTA, and MFFT algorithms are within 10% of the most efficient algorithms are 98%, 76%, 33%, and 20%, respectively.

The thrust of this report has been one-d.mensional DFT algorithms. However, two-dimensional DFT algorithms can be easily implemented with one-dimensional algorithms using the row/column technique. Using this procedure, one-dimensional DFT transforms are performed on all the rows, followed by one-dimensional transforms performed on all the columns of data resulting from the row transforms. Thus, the arithmetic requirements of a row/column implementation of an NxN DFT algorithm is simply 2N times the requirements of the selected N-point one-dimensional DFT algorithm.

True two-dimensional FFT algorithms have been developed which do not rely on one-dimensional tranforms [18]. These algorithms generally require less complex multiplications than the row/column methods. However, they are harder to implement and are less universal than the one-dimensional algorithms. Algorithms have also been developed for computing the two-dimensional DFT of arrays whose elements do not have rectangular spacing [19]. Refinement and extension of this work is very important to radar digital beam forming efforts as most phased array antennas have triangularly spaced elements. Although important, an in-depth examination of these algorithms is beyond the scope of this report.

The arithmetic requirements of a one-dimensional DFT are the same whether the coefficients are the standard ones of Equation (1) or those selected for individual custom responses. However, if the row/column method is selected for the two-dimensional DFT, only standard coefficients can be used. The custom shaping of the response of each transform output point afforded by the custom two-dimensional DFT requires a weighted sum of all the elements in the NxN data array. Each transform output point can have a unique NxN array of coefficients exhibiting none of the symmetrical properties of the standard DFT coefficients. Computing each custom DFT output point requires $4N^2$ real multiplications and $4N^2-2$ real additions. Computing all of the N^2 transform outputs therefore requires $4N^4$ real multiplications, $4N^4-2N^2$ real additions, and $8N^4-2N^2$ total real operati^ns.

The number of total real operations, TERM, real multiplications, and real additions for the custom DFT, DFT, radix-2 FFT, MFFT, SWIFT, WFTA, and PFA two-dimensional algorithms for array sizes 2x2 through 90x90 are shown in Tables 4-9 through 4-12. The arithmetic requirements shown in the tables are for the row/column method except for the custom DFT.

As the tables indicate, the arithmetic requirements for the two-dimensional custom DFT are enormous. However, if the number of desired customized transform output points is a small percentage of N^2, this algorithm can be useful. For example, if only four customized transform output points were required from an 8x8 data array, 2040 real operations would be required. To get four non-customized transform points from the radix-2 FFT would require the 1920 real operations needed to compute all the output points. The relative efficiency of the row/column algorithms is the same as the relative efficiency of the one-dimensional algorithms as the two-dimensional requirements are simply 2N times that of the one-dimensional requirements discussed earlier. As in the one-dimensional case, there are considerable differences in arithmetic complexity among the two-dimensional algorithms. For example, the 30x30 custom DFT, DFT, and WFTA algorithms require 6,478,200 real operations, 428,400 real operations, and 27,120 real operations, respectively. The differences are even greater for the larger arrays. For example, the 90x90 custom DFT, DFT, and PFA algorithms require 524,863,800 real operations, 11,631,600 real operations, and 369,360 real operations, respectively.

It is difficult to project the exact hardware size and cost for the various algorithms based solely on their arithmetic requirements. An analysis of the memory requirements, software complexity, optimum architectures, and availability of special purpose integrated circuits for each algorithm and array size is beyond the scope of this report. However, a brief review of present and near term arithmetic capabilities of digital integrated circuits will give insight into the feasibility of implementing the various algorithms for different array sizes.

Currently, TRW offers 8-bit and 16-bit multiplier/accumulator (MAC) chips which provide real multiplication and addition rates of 14 MHz and 9 MHz, respectively. These two TRW chips are packaged in dual-in-line packages with pin counts of 48 pins and 64 pins. Depending on the algorithm and required operating speeds, these chips can be multiplexed to reduce the total chip count.

Dramatic integrated circuit performance increases are expected in the near future as a result of the Department of Defense's Very High Speed Integrated Circuit (VHSIC) program. The $325 million, seven year long program which began in March of 1980, was designed to provide a fifty-fold improvement in high speed, high throughput signal and data processing integrated circuits. By the end of phase I of the program in mid-1984, six contractors will provide a pilot line production of chips with 1.25 micron architectural features, minimum throughput rates of 25 MHz, and a minimum functional throughput rate (FTR) of $5x10^{11}$ gate-Hz/cm^2. The pilot line production of chips with .5 to .8 micron architectural features, minimum throughput rates of 100 MHz, and a minimum FTR of 10^{13} gate-Hz/cm^2 will be required by the completion of phase II of the program in 1987 [20].

Several phase I VHSIC contractors will produce MAC chips. Preliminary reports indicate that IBM will produce a complex multiplier/accumulator (CMAC) chip. This implies a one-chip capability of performing a simultaneous set of approximately eight real operations (i.e., four real multiplications and four real additions) at a 25 MHz rate [21]. Westinghouse, another VHSIC contractor, plans to build a complex number arithmetic vector processor capable of performing 40 million complex number operations/second, which would only require two 6x8 in. printed circuit boards. In addition, Westinghouse is designing a ten-board array type processor capable of performing 200 million complex number operations/sec. or more than one billion real number operations/ sec. [22]. In addition to the VHSIC program, commercial very large scale integration (VLSI) chips produced with VHSIC technology are expected to provide VHSIC-like arithmetic capabilities.

A convenient way to compare the chip capabilities and algorithm requirements is to use the units: (1) millions of real multiplications/sec (MMPS), (2) millions of real additions/sec (MAPS), and (3) millions of total equivalent real multiplications/sec (TERMS). For example, the 16-bit TRW MAC chip is capable of 9 MMPS and 9 MAPS. The IBM VHSIC CMAC chip will offer roughly an eleven-fold improvement at 100 MMPS and 100 MAPS when developed. A hypothetical custom VHSIC/VLSI chip with at least 125 TERMS of arithmetic capability should be available by 1984. For the comparisons in this report, the time required to perform the transform will be arbitrarily assumed to be 1 sec. This choice of time makes the number of real multiplications, additions, and TERM found in the tables equal to the number of MMPS, MAPS, and TERMS, respectively. For example, computing a 64-point DFT in 1 sec requires 16,384 MMPS, 16,256 MAPS, and 20,448 TERMS. As the MMPS requirement of the 64-point DFT is more demanding than the MAPS requirements, the former dictates the use of 1,821 TRW MACs or 164 IBM CMACs. The TERMS numbers predict that 164 custom VHSIC/VLSI chips would be required.

Using the assumptions and methodology of the previous paragraph, Table 4-13 was constructed to estimate the relative number of TRW, IBM VHSIC, and custom VHSIC/VLSI chips required to meet the arithmetic requirements of various one and two-dimensional DFTs, radix-2 FFTs, and WFTAs. The chip count does not include non-arithmetic chips necessary for implementation, such as control and memory chips. However, a rough estimate of the number of required arithmetic chips can be found simply by scaling the chip count in the table by the ratio of 1 μsec and the desired transform time. This table summarizes the relative differences of complexity among the various algorithms and suitability of current and proposed hardware. For example, the DFT shown in the table requires more arithmetic chips than any other algorithm except the custom two-dimensional DFT. As shown in the table, the custom VHSIC/VLSI chip offers no significant reductions in the DFT chip count. This illustrates that the TRW and IBM chips are well suited to the DFT. In contrast, the WFTA requires fewer arithmetic chips, although it is not particularly well suited to the TRW and IBM chips. Roughly a three-fold improvement is gained using custom VHSIC/VLSI chips tailored to the WFTA's required multiplication to addition ratio. The radix-2 FFT compares surprisingly well with the WFTA when implemented with the MAC and CMAC chips of TRW and IBM. The extra arithmetic chips required by the radix-2 FFT would probably be offset by the extra control and memory chips required by the more structurally complex WFTA algorithm. However, if custom chips are available, the radix-2 FFT would generally require 200% to 300% of the arithmetic chips required by the WFTA. The PFA and radix-4 FFT algorithms are not shown in the table as they are very close to the numbers given for the WFTA and radix-2 algorithms, respectively. Likewise, the MFFT and SWIFT algorithms are not shown as they reside between the radix-2 FFT and the WFTA in performance.

TABLE 4-1. 1D REAL OPERATIONS REQUIREMENTS

N	DFT	RADIX 2	RADIX 4	MFFT	SWIFT	WFTA	PFA
		FFT	FFT				
2	28	4	---	4	4	8	4
3	66	---	---	16	16	18	16
4	120	40	34	16	16	24	16
5	190	---	---	48	48	46	44
6	276	---	---	56	44	54	44
7	378	---	---	120	96	90	88
8	496	120	---	66	56	68	56
9	630	---	---	120	132	110	104
10	780	---	---	140	116	108	108
12	1,128	---	---	148	112	114	112
14	1,540	---	---	286	220	204	204
15	1,770	---	---	272	224	196	212
16	2,016	320	272	182	168	184	168
18	2,556	---	---	324	300	252	252
20	3,160	---	---	344	272	258	256
21	3,486	---	---	508	400	352	376
24	4,560	---	---	410	296	292	296
28	6,216	---	---	646	496	466	464
30	7,140	---	---	688	508	452	484
32	8,128	800	---	518	---	---	---
35	9,730	---	---	1,008	816	772	748
36	10,296	---	---	768	672	578	576
40	12,720	---	---	882	664	620	632
42	14,028	---	---	1,202	884	788	836
45	16,110	---	---	1,224	1,092	944	936
48	18,336	---	---	982	760	734	760
56	24,976	---	---	1,548	1,160	1,076	1,096
60	28,680	---	---	1,580	1,136	1,026	1,088
63	31,626	---	---	2,064	1,788	1,604	1,548
64	32,640	1,920	1,632	1,254	---	---	---
70	39,060	---	---	2,342	1,772	1,684	1,636
72	41,328	---	---	1,890	1,560	1,340	1,368
80	51,040	---	---	2,038	1,608	1,558	1,544
84	56,280	---	---	2,674	1,936	1,746	1,840
90	64,620	---	---	2,364	2,364	2,068	2,052

TABLE 4-2. 1D REAL OPERATIONS REQUIREMENTS

N	DFT	RADIX 2 FFT	RADIX 4 FFT	MFFT	SWIFT	WFTA	PFA
105	87,990	---	---	3,956	3,008	2,740	2,804
112	100,128	---	---	3,464	2,712	2,646	2,584
120	114,960	---	---	2,634	2,632	2,356	2,536
126	126,756	---	---	4,734	3,828	3,460	3,348
128	130,816	4,480	---	3,142	---	---	---
140	156,520	---	---	5,150	3,824	3,650	3,552
144	165,600	---	---	4,278	3,624	3,302	3,240
168	225,456	---	---	6,164	4,376	3,916	4,184
180	258,840	---	---	6,408	5,088	4,498	4,464
210	352,380	---	---	8,938	6,436	5,900	6,028
240	460,320	---	---	8,342	6,104	5,654	5,912
252	507,528	---	---	10,326	8,160	7,426	7,200
256	523,776	10,240	8,704	7,174	---	---	---
280	626,640	---	---	11,676	8,488	8,004	7,944
315	793,170	---	---	14,760	11,964	11,592	10,512
336	902,496	---	---	13,480	9,928	9,302	9,544
360	1,036,080	---	---	14,610	11,256	9,900	10,008
420	1,410,360	---	---	19,322	13,712	12,642	12,896
504	2,031,120	---	---	23,148	17,832	16,116	15,912
512	2,096,128	23,040	---	16,902	---	---	---
560	2,507,680	---	---	25,288	18,936	19,102	17,848
630	3,173,940	---	---	32,646	25,188	24,444	22,284
720	4,145,760	---	---	31,734	25,032	23,678	22,536
840	5,643,120	---	---	42,820	29,944	27,388	28,312
1,008	8,126,496	---	---	49,800	39,192	38,222	35,352
1,024	8,386,560	51,200	43,520	37,382	---	---	---
1,260	12,698,280	---	---	69,678	52,896	51,410	47,088
1,680	22,575,840	---	---	91,496	65,768	64,046	62,504
2,048	33,550,336	112,640	---	84,998	---	---	---
2,520	50,798,160	---	---	151,932	113,352	109,124	101,736
4,096	134,209,536	245,760	208,896	184,326	---	---	---
5,040	203,202,720	---	---	321,480	242,344	255,302	221,112

TABLE 4-3. 1D TOTAL EQUIVALENT REAL MULTIPLICATIONS (TERM) REQUIREMENTS

N	DFT	RADIX 2 FFT	RADIX 4 FFT	MFFT	SWIFT	WFTA	PFA
2	19	1	---	1	1	5	1
3	44	---	---	7	7	9	7
4	78	22	18	4	4	12	4
5	123	---	---	24	24	21	19
6	177	---	---	26	17	27	17
7	242	---	---	66	51	36	34
8	316	66	---	26	17	29	17
9	401	---	---	60	66	44	41
10	495	---	---	71	53	42	42
12	714	---	---	67	40	42	40
14	973	---	---	153	109	75	75
15	1,118	---	---	143	107	75	91
16	1,272	176	140	73	60	73	57
18	1,611	---	---	165	141	93	93
20	1,990	---	---	170	116	96	94
21	2,195	---	---	274	202	127	151
24	2,868	---	---	196	107	103	107
28	3,906	---	---	333	232	166	164
30	4,485	---	---	364	229	164	196
32	5,104	440	---	229	---	---	---
35	6,108	---	---	552	423	273	300
36	6,462	---	---	384	300	206	204
40	7,980	---	---	446	277	221	233
42	8,799	---	---	646	425	275	323
45	10,103	---	---	660	546	334	377
48	11,496	---	---	465	292	257	283
56	15,652	---	---	801	527	371	391
60	17,970	---	---	821	488	360	422
63	19,814	---	---	1,122	921	548	600
64	20,448	1,056	840	557	---	---	---
70	24,465	---	---	1,279	881	580	634
72	25,884	---	---	962	681	461	489
80	31,960	---	---	1,017	684	544	581
84	35,238	---	---	1,410	892	594	688
90	40,455	---	---	1,563	1,137	712	798

TABLE 4-4. 1D TOTAL EQUIVALENT REAL MULTIPLICATIONS (TERM) REQUIREMENTS

N	DFT	RADIX 2 FFT	RADIX 4 FFT	MFFT	SWIFT	WFTA	PFA
105	55,073	---	---	2,180	1,514	927	1,144
112	62,664	---	---	1,766	1,236	897	943
120	71,940	---	---	1,328	1,111	799	979
126	79,317	---	---	2,573	·1,905	1,159	1,263
128	81,856	2,464	---	1,461	---	---	---
140	97,930	---	---	2,767	1,832	1,232	1,338
144	103,608	---	---	2,133	1,596	1,115	1,185
168	141,036	---	---	3,263	1,973	1,297	1,565
180	161,910	---	---	3,414	2,364	1,516	1,686
210	220,395	---	---	4,920	3,133	1,958	2,392
240	287,880	---	---	4,321	2,612	1,892	2,303
252	317,394	---	---	5,537	3,936	2,446	2,652
256	327,552	5,632	4,480	3,333	---	---	---
280	391,860	---	---	6,285	3,979	2,643	2,991
315	495,968	---	---	8,184	6,117	3,788	4,166
336	564,312	---	---	7,054	4,492	3,047	3,613
360	647,820	---	---	7,814	5,133	3,261	3,777
420	881,790	---	---	10,504	6,476	4,128	4,994
504	1,269,828	---	---	12,441	8,439	5,211	5,871
512	1,310,464	12,672	---	8,069	---	---	---
560	1,567,720	---	---	13,462	8,868	6,226	6,787
630	1,984,185	---	---	13,083	12,549	7,890	8,646
720	2,591,640	---	---	16,793	11,436	7,694	8,589
840	3,527,580	---	---	23,299	13,897	8,785	10,933
1008	5,079,816	---	---	26,502	18,516	12,221	13,191
1024	5,242,368	28,160	22,400	17,797	---	---	---
1260	7,937,370	---	---	33,195	25,728	16,412	17,922
1680	14,111,160	---	---	49,310	30,524	20,378	24,281
2048	20,970,496	61,952	---	41,221	---	---	---
2520	31,750,740	---	---	83,301	54,291	34,403	38,679
4096	83,884,032	135,168	107,520	89,093	---	---	---
5040	127,005,480	---	---	174,774	114,772	79,856	84,603

TABLE 4-5. 1D REAL MULTIPLICATION REQUIREMENTS

N	FACTORS	DFT	RADIX 2 FFT	RADIX 4 FFT	MFFT	SWIFT	WFTA	PFA
2	2	16	0	---	0	0	4	0
3	3	36	---	---	4	4	6	4
4	4	64	16	12	0	0	8	0
5	5	100	---	---	16	16	12	10
6	2x3	144	---	---	16	8	18	8
7	7	196	---	---	48	36	18	16
8	8	256	48	---	12	4	16	4
9	9	324	---	---	40	44	22	20
10	5x2	400	---	---	48	32	20	20
12	3x4	576	---	---	40	16	18	16
14	7x2	784	---	---	108	72	32	32
15	5x3	900	---	---	100	68	34	50
16	16	1,024	128	96	36	24	36	20
18	9x2	1,296	---	---	112	88	40	40
20	5x4	1,600	---	---	112	64	42	40
21	7x3	1,764	---	---	196	136	52	76
24	3x8	2,304	---	---	124	44	40	44
28	7x4	3,136	---	---	228	144	66	64
30	5x3x2	3,600	---	---	256	136	68	100
32	2^5	4,096	320	---	132	---	---	---
35	5x7	4,900	---	---	400	292	106	150
36	9x4	5,184	---	---	256	176	82	80
40	5x8	6,400	---	---	300	148	88	100
42	7x3x2	7,056	---	---	460	272	104	152
45	5x9	8,100	---	---	472	364	130	190
48	16x3	9,216	---	---	292	136	98	124
56	7x8	12,544	---	---	552	316	136	156
60	5x4x3	14,400	---	---	568	272	138	200
63	7x9	15,876	---	---	808	632	196	284
64	2^6	16,384	768	576	324	---	---	---
70	5x7x2	19,600	---	---	924	584	212	300
72	9x8	20,736	---	---	652	388	168	196
80	5x16	25,600	---	---	676	376	206	260
84	7x4x3	28,224	---	---	988	544	210	304
90	5x9x2	32,400	---	---	1,120	728	260	380

TABLE 4-6. 1D REAL MULTIPLICATION REQUIREMENTS

N	FACTORS	DFT	RADIX 2 FFT	RADIX 4 FFT	MFFT	SWIFT	WFTA	PFA
105	5x7x3	44,100	---	---	1,588	1,016	322	590
112	7x16	50,176	---	---	1,200	744	314	396
120	5x8x3	57,600	---	---	892	604	280	460
126	7x9x2	63,504	- -	---	1,852	1,264	392	568
128	2^7	65,536	1,792	---	900	---	---	---
140	5x7x4	78,400	---	---	1,972	1,168	426	600
144	9x16	82,944	---	---	1,444	920	386	500
168	7x8x3	112,896	---	---	2,296	1,172	424	692
180	5x9x4	129,600	---	---	2,416	1,456	522	760
210	5x7x2x3	176,400	---	---	3,580	2,032	644	1,180
240	5x16x3	230,400	---	---	2,980	1,448	638	1,100
252	7x9x4	254,016	---	---	3,940	2,528	786	1,136
256	2^8	262,144	4,096	3,072	2,052	---	---	---
280	5x7x8	313,600	---	---	4,488	2,476	856	1,340
315	5x7x9	396,900	---	---	5,992	4,168	1,186	2,050
336	7x16x3	451,584	---	---	4,912	2,680	962	1,636
360	5x9x8	518,400	---	---	5,548	3,092	1,048	1,700
420	5x7x3x4	705,600	---	---	7,564	4,064	1,290	2,360
504	7x9x8	1,016,064	---	---	8,872	5,308	1,576	2,524
512	2^9	1,048,576	9,216	---	5,124	---	---	---
560	5x7x16	1,254,400	---	---	9,520	5,512	1,934	3,100
630	5x7x9x12	1,587,600	---	---	13,228	8,336	2,372	4,100
720	5x9x16	2,073,600	---	---	11,812	6,904	2,366	3,940
840	5x7x8x3	2,822,400	---	---	16,792	8,548	2,584	5,140
1,008	7x9x16	4,064,256	---	---	18,736	11,624	3,554	5,804
1,024	2^{10}	4,194,304	20,480	15,360	11,268	---	---	---
1,260	5x7x9x4	6,350,400	---	---	27,700	16,672	4,746	8,200
1,680	5x7x16x3	11,289,600	---	---	35,248	18,776	5,822	11,540
2,048	2^{11}	16,777,216	45,056	---	26,628	---	---	---
2,520	5x7x9x8	25,401,600	---	---	60,424	34,604	9,496	17,660
4,096	2^{12}	67,108,864	98,304	73,728	57,348	---	---	---
5,040	5x7x9x16	101,606,400	---	---	125,872	72,248	21,374	39,100

TABLE 4-7. 1D REAL ADDITION REQUIREMENTS

N	FACTORS	DFT	RADIX 2 FFT	RADIX 4 FFT	MFFT	SWIFT	WFTA	PFA
2	2	12	4	---	4	4	4	4
3	3	30	---	---	12	12	12	12
4	4	56	24	22	16	16	16	16
5	5	90	---	---	32	32	34	34
6	2x3	132	---	---	40	36	36	36
7	7	182	---	---	72	60	72	72
8	8	240	72	---	54	52	52	52
9	9	306	---	---	80	88	88	84
10	5x2	380	---	---	92	84	88	88
12	3x4	552	---	---	108	96	96	96
14	7x2	756	---	---	178	148	172	172
15	5x3	870	---	---	172	156	162	162
16	16	992	192	176	146	144	148	148
18	9x2	1,260	---	---	212	212	212	212
20	5x4	1,560	---	---	232	208	216	216
21	7x3	1,722	---	---	312	264	300	300
24	3x8	2,256	---	---	286	252	252	252
28	7x4	3,080	---	---	418	352	400	400
30	5x3x2	3,540	---	---	432	372	384	384
32	2^5	4,032	480	---	386	---	---	---
35	5x7	4,830	---	---	608	524	666	598
36	9x4	5,112	---	---	512	496	496	496
40	5x8	6,320	---	---	582	516	532	532
42	7x3x2	6,972	---	---	742	612	684	684
45	5x9	8,010	---	---	752	728	814	746
48	16x3	9,120	---	---	690	624	636	636
56	7x8	12,432	---	---	996	844	940	940
60	5x4x3	14,280	---	---	1,012	864	888	888
63	7x9	15,750	---	---	1,256	1,156	1,408	1,264
64	2^6	16,256	1,152	1,056	930	---	---	---
70	5x7x2	19,460	---	---	1,418	1,188	1,472	1,336
72	9x8	20,592	---	---	1,238	1,172	1,172	1,172
80	5x16	25,440	---	---	1,362	1,232	1,352	1,284
84	7x4x3	28,056	---	---	1,686	1,392	1,536	1,536
90	5x9x2	32,220	---	---	1,772	1,636	1,808	1,672

TABLE 4-8. 1D REAL ADDITION REQUIREMENTS

			RADIX 2	RADIX 4				
N	FACTORS	DFT	FFT	FFT	MFFT	SWIFT	WFTA	PFA
105	5x7x3	43,890	---	---	2,368	1,992	2,418	2,214
112	7x16	49,952	---	---	2,264	1,968	2,332	2,188
120	5x8x3	57,360	---	---	1,742	2,028	2,076	2,076
126	7x9x2	63,252	---	---	2,882	2,564	3,068	2,780
128	2^7	65,280	2,688	---	2,242	---	---	---
140	5x7x4	78,120	---	---	3,178	2,656	3,224	2,952
144	9x16	82,656	---	---	2,834	2,704	2,916	2,740
168	7x8x3	112,560	---	---	3,868	3,204	3,492	3,492
180	5x9x4	129,240	---	---	3,992	3,632	3,976	3,704
210	5x7x2x3	175,980	---	---	5,358	4,404	5,256	4,848
240	5x16x3	229,920	---	---	5,362	4,656	5,016	4,812
252	7x9x4	253,512	---	---	6,386	5,632	6,640	6,064
256	2^8	261,632	6,144	5,632	5,122	---	---	---
280	5x7x8	313,040	---	---	7,188	6,012	7,148	6,604
315	5x7x9	396,270	---	---	8,768	7,796	10,406	8,462
336	7x16x3	450,912	---	---	8,568	7,248	8,340	7,908
360	5x9x8	517,680	---	---	9,062	8,164	8,852	8,308
420	5x7x3x4	704,760	---	---	11,758	9,648	11,352	10,536
504	7x9x8	1,015,056	---	---	14,276	12,524	14,540	13,388
512	2^9	1,047,552	13,824	---	11,778	---	---	---
560	5x7x16	1,253,280	---	---	15,768	13,424	17,168	14,748
630	5x7x9x2	1,586,340	---	---	19,418	16,852	22,072	18,184
720	5x9x16	2,072,160	---	---	19,922	18,128	21,312	18,596
840	5x7x8x3	2,820,720	---	---	26,028	21,396	24,804	23,172
1,008	7x9x16	4,062,240	---	---	31,064	27,568	34,668	29,548
1,024	2^{10}	4,192,256	30,720	28,160	26,114	---	---	---
1,260	5x7x9x4	6,347,880	---	---	41,978	36,224	46,664	38,888
1,680	5x7x16x3	11,286,240	---	---	56,248	46,992	58,224	50,964
2,048	2^{11}	16,773,120	67,584	---	58,370	---	---	---
2,520	5x7x9x8	25,396,560	---	---	91,508	78,748	99,628	84,076
4,096	2^{12}	67,100,672	147,456	135,168	126,978	---	---	---
5,040	5x7x9x16	101,596,320	---	---	195,608	170,096	233,928	182,012

TABLE 4-9. 2D REAL OPERATIONS REQUIREMENTS

N	N^2	DFT (CUSTOM)	DFT	RADIX 2 FFT	MFFT	SWIFT	WFTA	PFA
2	4	120	112	16	16	16	32	16
3	9	630	396	---	96	96	108	96
4	16	2,016	960	320	128	128	192	128
5	25	4,950	1,900	---	480	480	460	440
6	36	10,296	3,312	---	672	528	648	528
7	49	19,110	5,292	---	1,680	1,344	1,260	1,232
8	64	32,640	7,936	1,920	1,056	896	1,088	896
9	81	52,326	11,340	---	2,160	2,376	1,980	1,872
10	100	79,800	15,600	---	2,800	2,320	2,160	2,160
12	144	165,600	27,072	---	3,552	2,688	2,736	2,688
14	196	306,936	43,120	---	8,008	6,160	5,712	5,712
15	225	404,550	53,100	---	8,160	6,720	5,880	6,360
16	256	523,776	64,512	10,240	5,824	5,376	5,888	5,376
18	324	839,160	92,016	---	11,664	10,800	9,072	9,072
20	400	1,279,200	126,400	---	13,760	10,880	10,320	10,240
21	441	1,554,966	146,412	---	21,336	16,800	14,784	15,792
24	576	2,653,056	218,880	---	19,680	14,208	14,016	14,208
28	784	4,915,680	348,096	---	36,176	27,776	26,096	25,984
30	900	6,478,200	428,400	---	41,280	30,[illegible]80	27,120	29,040
32	1,024	8,386,560	520,192	51,200	33,152	---	---	---
35	1,225	12,002,550	681,100	---	70,560	57,120	54,040	52,360
36	1,296	13,434,336	741,312	---	55,296	48,384	41,616	41,472
40	1,600	20,476,800	1,017,600	---	70,560	53,120	49,600	50,560
42	1,764	24,890,040	1,178,352	---	100,968	74,256	66,192	70,224
45	2,025	32,800,950	1,449,900	---	110,160	98,280	84,960	84,240
48	2,304	42,462,720	1,760,256	---	94,272	72,960	70,464	72,960
56	3,136	78,669,696	2,797,312	---	173,376	129,920	120,512	122,752
60	3,600	103,672,800	3,441,600	---	189,600	136,320	123,120	130,560
63	3,969	126,015,750	3,984,876	---	260,064	225,288	202,104	195,048
64	4,096	134,209,536	4,177,920	245,760	160,512	---	---	---
70	4,900	192,070,200	5,468,400	---	327,880	248,080	235,760	229,040
72	5,184	214,980,480	5,951,232	---	272,160	224,640	192,960	196,992
80	6,400	327.667,200	8,166,400	---	326,080	257,280	249,280	247,040
84	7,056	398,282,976	9,455,040	---	449,232	325,248	293,328	309,120
90	8,100	524,863,800	11,631,600	---	520,560	425,520	372,240	369,360

TABLE 4-10. 2D TOTAL EQUIVALENT REAL MULTIPLICATIONS (TERM) REQUIREMENTS

N	N^2	DFT (CUSTOM)	DFT	RADIX 2 FFT	MFFT	SWIFT	WFTA	PFA
2	4	78	76	4	4	4	20	4
3	9	401	261	---	42	42	54	42
4	16	1,272	624	176	32	32	96	32
5	25	3,113	1,225	---	240	240	210	190
6	36	6,462	2,124	---	312	204	324	204
7	49	11,981	3,381	---	924	714	504	476
8	64	20,448	5,056	1,056	416	272	464	272
9	81	32,765	7,209	---	1,080	1,188	792	738
10	100	49,950	9,900	---	1,420	1,060	840	840
12	144	103,608	17,136	---	1,608	960	1,008	960
14	196	191,982	27,244	---	4,284	3,052	2,100	2,100
15	225	253,013	33,525	---	4,290	3,210	2,250	2,730
16	256	327,552	40,704	5,632	2,336	1,920	2,336	1,824
18	324	524,718	57,996	---	5,940	5,076	3,348	3,348
20	400	799,800	79,600	---	6,800	4,640	3,840	3,760
21	441	972,185	92,169	---	11,508	8,484	5,334	6,342
24	576	1,658,592	137,664	---	9,408	5,136	4,944	5,136
28	784	3,072,888	218,736	---	18,648	12,992	9,296	9,184
30	900	4,049,550	269,100	---	21,840	13,740	9,840	11,760
32	1,024	5,242,368	326,656	28,160	14,656	---	---	---
35	1,225	7,502,513	427,525	---	38,640	29,610	19,110	21,000
36	1,296	8,397,432	465,264	---	27,648	21,600	14,832	14,688
40	1,600	12,799,200	638,400	---	35,680	22,160	17,680	18,640
42	1,764	15,557,598	739,116	---	54,264	35,700	23,100	27,132
45	2,025	20,502,113	909,225	---	59,400	49,140	30,060	33,930
48	2,304	26,540,928	1,103,616	---	44,640	28,032	24,672	27,168
56	3,136	49,170,912	1,753,024	---	89,712	59,024	41,552	43,792
60	3,600	64,798,200	2,156,400	---	98,520	58,560	43,200	50,640
63	3,969	78,762,821	2,496,501	---	141,372	116,046	69,048	75,600
64	4,096	83,884,032	2,617,344	138,168	71,296	---	---	---
70	4,900	120,047,550	3,425,100	---	179,060	123,340	81,200	88,760
72	5,184	134,366,688	3,727,296	---	138,528	98,064	66,384	70,416
80	6,400	204,796,800	5,113,600	---	162,720	109,440	87,040	92,960
84	7,056	248,932,152	5,919,984	---	236,880	149,856	99,792	115,584
90	8,100	328,045,950	7,281,900	---	281,340	204,660	128,160	143,640

TABLE 4-11. 2D REAL MULTIPLICATION REQUIREMENTS

N	N^2	DFT (CUSTOM)	DFT	RADIX 2 FFT	MFFT	SWIFT	WFTA	PFA
2	4	64	64	0	0	0	16	0
3	9	324	216	---	24	24	36	24
4	16	1,024	512	128	0	0	-64	0
5	25	2,500	1,000	---	160	160	120	100
6	36	5,184	1,728	---	192	96	216	96
7	49	9,604	2,744	---	672	504	252	224
8	64	16,384	4,096	768	192	64	256	64
9	81	26,244	5,832	---	720	792	396	360
10	100	40,000	8,000	---	960	640	400	400
12	144	82,944	13,824	---	960	384	432	384
14	196	153,664	21,952	---	3,024	2,016	896	896
15	225	202,500	27,000	---	3,000	2,040	1,020	1,500
16	256	262,144	32,768	4,096	1,152	768	1,152	640
18	324	419,904	46,656	---	4,032	3,168	1,440	1,440
20	400	640,000	64,000	---	4,480	2,560	1,680	1,600
21	441	777,924	74,088	---	8,232	5,712	2,184	3,192
24	576	1,327,104	110,592	---	5,952	2,112	1,920	2,112
28	784	2,458,624	175,616	---	12,768	8,064	3,696	3,584
30	900	3,240,000	216,000	---	15,360	8,160	4,080	6,000
32	1,024	4,194,304	262,144	20,480	8,448	---	---	---
35	1,225	6,002,500	343,000	---	28,000	20,440	7,420	10,500
36	1,296	6,718,464	373,248	---	18,432	12,672	5,904	5,760
40	1,600	10,240,000	512,000	---	24,000	11,840	7,040	8,000
42	1,764	12,446,784	592,704	---	38,640	22,848	8,736	12,768
45	2,025	16,402,500	729,000	---	42,480	32,760	11,700	17,100
48	2,304	21,233,664	884,736	---	28,032	13,056	9,408	11,904
56	3,136	39,337,984	1,404,928	---	61,824	35,392	15,232	17,472
60	3,600	51,840,000	1,728,000	---	68,160	32,640	16,560	24,000
63	3,969	63,011,844	2,000,376	---	101,808	79,632	24,696	35,784
64	4,096	67,108,864	2,097,152	98,304	41,472	---	---	---
70	4,900	96,040,000	2,744,000	---	129,360	81,760	29,680	42,000
72	5,184	107,495,424	2,985,984	---	93,888	55,872	24,192	28,224
80	6,400	163,840,000	4,096,000	---	108,160	60,160	32,960	41,600
84	7,056	199,148,544	4,741,632	---	165,984	91,392	35,280	51,072
90	8,100	262,440,000	5,832,000	---	201,600	131,040	46,800	68,400

TABLE 4-12. 2D REAL ADDITION REQUIREMENTS

N	N^2	DFT (CUSTOM)	DFT	RADIX 2 FFT	MFFT	SWIFT	WFTA	PFA
2	4	56	48	16	16	16	16	16
3	9	306	180	---	72	72	72	72
4	16	992	448	192	128	128	128	128
5	25	2,450	900	---	320	320	340	340
6	36	5,112	1,584	---	480	432	432	432
7	49	9,506	2,548	---	1,008	840	1,008	1,008
8	64	16,256	3,840	1,152	864	832	832	832
9	81	26,082	5,508	---	1,440	1,584	1,584	1,512
10	100	39,800	7,600	---	1,840	1,680	1,760	1,760
12	144	82,656	13,248	---	2,592	2,304	2,304	2,304
14	196	153,272	21,168	---	4,984	4,144	4,816	4,816
15	225	202,050	26,100	---	5,160	4,680	4,860	4,860
16	256	261,632	31,744	6,144	4,672	4,608	4,736	4,736
18	324	419,256	45,360	---	7,632	7,632	7,632	7,632
20	400	639,200	62,400	---	9,280	8,320	8,640	8,640
21	441	777,042	72,324	---	13,104	11,088	12,600	12,600
24	576	1,325,952	108,288	---	13,728	12,096	12,096	12,096
28	784	2,457,056	172,480	---	23,408	19,712	22,400	22,400
30	900	3,238,200	212,400	---	25,920	22,320	23,040	23,040
32	1,024	4,192,256	258,048	30,720	24,704	---	---	---
35	1,225	6,000,050	338,100	---	42,560	36,680	46,620	41,860
36	1,296	6,715,872	368,064	---	36,864	35,712	35,712	35,712
40	1,600	10,236,800	505,600	---	46,560	41,280	42,560	42,560
42	1,764	12,443,256	535,648	---	62,328	51,408	57,456	57,456
45	2,025	16,398,450	720,900	---	67,680	65,520	73,260	67,140
48	2,304	21,229,056	875,520	---	66,240	59,904	61,056	61,056
56	3,136	39,331,712	1,392,384	---	111,552	94,528	105,280	105,280
60	3,600	51,832,800	1,713,600	---	121,440	103,680	106,560	106,560
63	3,969	63,003,906	1,984,500	---	158,256	145,656	177,408	159,264
64	4,096	67,100,672	2,080,768	147,456	119,040	---	---	---
70	4,900	96,030,200	2,724,400	---	198,520	166,320	206,080	187,040
72	5,184	107,485,056	2,965,248	---	178,272	168,768	168,768	168,768
80	6,400	163,827,200	4,070,400	---	217,920	197,120	216,320	205,440
84	7,056	199,134,432	4,713,408	---	283,248	233,856	258,048	258,048
90	8,100	262,423,800	5,799,600	---	318,960	294,480	325,440	300,960

TABLE 4-13. APPROXIMATE ARITHMETIC CHIP COUNT FOR 1uSEC TRANSFORM

	TRW			VHSIC IBM			CUSTOM VHSIC/VLSI		
N	DFT	FFT(2)	WFTA	DFT	FFT(2)	WFTA	DFT	FFT(2)	WFTA
8	29	8	6	3	1	1	3	1	1
16	114	22	17	11	2	2	11	2	1
24	256	---	28	23	---	3	23	---	1
30	400	---	43	36	---	4	36	---	2
32	456	54	---	41	5	---	41	4	---
48	1,024	---	71	93	---	7	92	---	2
60	1,600	---	99	144	---	9	144	---	3
64	1,821	128	---	164	12	---	164	9	---
72	2,304	---	131	208	---	12	207	---	4
90	3,600	---	201	324	---	18	324	---	6
120	6,400	---	231	576	---	21	576	---	7
128	7,282	299	---	656	27	---	655	20	---
144	9,216	---	324	830	---	30	829	---	9
180	14,400	---	442	1,296	---	40	1,296	---	13
256	29,128	683	---	2,622	62	---	2,621	45	---
360	57,600	---	984	5,184	---	89	5,183	---	26
504	112,896	---	1,616	10,161	---	146	10,159	---	42
512	116,509	1,536	---	10,486	139	---	10,484	102	---
720	230,400	---	2,368	20,736	---	214	20,734	---	62
1,008	451,584	---	3,852	40,643	---	347	40,639	---	98
1,024	466,034	3,414	---	41,943	308	---	41,939	225	---
1,680	1,254,400	---	6,470	112,896	---	583	112,890	---	163
2.048	1,864,136	7,510	---	167,773	676	---	167,764	496	---
2,520	2,822,400	---	11,070	254,016	---	997	254,006	---	276
4,096	7,456,541	16,384	---	671,089	1,475	---	671,073	1,082	---
5,040	11,289,600	---	25,992	1,016,064	---	2,340	1,016,044	---	639
8x8	456	128	93	41	12	9	41	9	4
16x16	3,641	683	527	328	62	48	326	45	19
24x24	12,288	---	1,344	1,106	---	121	1,102	---	40
32x32	29,128	3,414	---	2,622	308	---	2,614	226	---
48x48	98,304	---	6,784	8,848	---	611	8,829	---	198
60x60	192,000	---	11,840	17,280	---	1,066	17,252	---	346
64x64	233,017	16,384	---	20,972	1,475	---	20,939	1,082	---
90x90	648,000	---	36,160	58,320	---	3,255	58,256	---	1,026

Appendix A. WFTA SHORT DFT ALGORITHMS

Algorithms are given to compute the DFT for lengths 2, 3, 4, 5, 7, 8, 9 and 16. These algorithms were taken from [23], but have been credited to the work of Radar and Winograd [23,13]. The complex input data x(1), x(2),..., x(N) and complex output data X(1), X(2),..., X(N) are in natural order. The complex values M1, M2,..., MM are the results of the M complex multiplications required for the small point transform. The complex values T1, T2,... and S1, S2,... are temporary values derived from the input data and intermediate results, respectively. Generally, the operations must be performed in the order listed. The total number of trivial and non-trivial complex multiplications and additions required for each DFT is listed with the algorithm. In addition, the number of complex multiplications by W^0 or "1" is given in parentheses.

(1) N=2; 2 complex multiplications (2), 2 complex additions.

```
M1=1*(x(1)+x(2))
M2=1*(x(1)-x(2))
X(1)=M1
X(2)=M2
```

(2) N=3; 3 complex multiplications (1), 6 complex additions, u=2π/3.

```
Coefficients:  C1=-3/2
               C2=jsin u

T1=x(2)+x(3)
M1=1*(x(1)+T1)
M2=C1*T1
M3=C2*(x(3)-x(2))
S1=M1+M2
X(1)=M1
X(2)=S1+M3
X(3)=S1-M3
```

(3) N=4; 4 complex multiplications (3), 8 complex additions.

```
T1=x(1)+x(3)
T2=x(2)+x(4)
M1=1*(T1+T2)
M2=1*(T1-T2)
M3=1*(x(1)-x(3))
M4=j*(x(4)-x(2)`
X(1)=M1
X(2)=M3+M4
X(3)=M2
X(4)=M3-M4
```

(4) N=5; 6 complex multiplications (1), 17 complex additions, u=2π/5.

```
Coefficients:  C1=-5/4
               C2=(cos u-cos 2u)/2
               C3=-j sin u
               C4=-j(sin u+sin 2u)
               C5=j(sin u-sin 2u)

T1=x(2)+x(5)
T2=x(3)+x(4)
T3=x(2)-x(5)
T4=x(4)-x(3)
T5=T1+T2
M1=1*(x(1)+T5)
M2=C1*T5
M3=C2*(T1-T2)
M4=C3*(T3+T4)
M5=C4*T4
M6=C5*T3
S1=M1+M2
S2=S1+M3
S3=M4-M5
S4=S1-M3
S5=M4+M6
X(1)=M1
X(2)=S2+S3
X(3)=S4+S5
X(4)=S4-S5
X(5)=S2-S3
```

(5) N=7; 9 complex multiplications (1), 36 complex additions, u=2π/7.

```
Coefficients:  C1=-7/6
               C2=(2cos u-cos 2u-cos 3u)/3
               C3=(cos u-2cos 2u+cos 3u)/3
               C4=(cos u+cos 2u-2cos 3u)/3
               C5=-j(sin u+sin 2u-sin 3u)/3
               C6=j(2sin u-sin 2u+sin 3u)/3
               C7=j(sin u-2sin 2u-sin 3u)/3
               C8=j(sin u+sin 2u+2sin 3u)/3

T1=x(2)+x(7)
T2=x(3)+x(6)
T3=x(4)+x(5)
T4=T1+T2+T3
T5=x(2)-x(7)
T6=x(3)-x(6)
T7=x(5)-x(4)
T8=T1-T3
T9=T3-T2
T10=T5+T6+T7
T11=T7-T5
T12=T6-T7
T13=-T8-T9
T14=-T11-T12
M1=1*(x(1)+T4)
M2=C1*T4
M3=C2*T8
M4=C3*T9
```

```
M5=C4*T13
M6=C5*T10
M7=C6*T11
M8=C7*T12
M9=C8*T14
S1=-M3-M4
S2=-M3-M5
S3=-M7-M8
S4=M7+M9
S5=M1+M2
S6=S5-S1
S7=S5+S2
S8=S5+S1-S2
S9=M6-S3
S10=M6-S4
S11=M6+S3+S4
X(1)=M1
X(2)=S6+S9
X(3)=S7+S10
X(4)=S8-S11
X(5)=S8+S11
X(6)=S7-S10
X(7)=S6-S9
```

(6) N=8; 8 complex multiplications (4), 26 complex additions, u=2π/8.

```
Coefficients:  C1=cos u
               C2=-jsin u

T1=x(1)+x(5)
T2=x(3)+x(7)
T3=x(2)+x(6)
T4=x(2)-x(6)
T5=x(4)+x(8)
T6=x(4)-x(8)
T7=T1+T2
T8=T3+T5
M1=1*(T7+T8)
M2=1*(T7-T8)
M3=1*(T1-T2)
M4=1*(x(1)-x(5))
M5=C1*(T4-T6)
M6=j*(T5-T3)
M7=j*(x(7)-x(3))
M8=C2*(T4+T6)
S1=M4+M5
S2=M4-M5
S3=M7+M8
S4=M7-M8
X(1)=M1
X(2)=S1+S3
X(3)=M3+M6
X(4)=S2-S4
X(5)=M2
```

```
X(6)=S2+S4
X(7)=M3-M6
X(8)=S1-S3
```

(7) N=9; 11 complex multiplications (1), 44 complex additions, $u=2\pi/9$.

```
Coefficients:  C1=3/2
               C2=-1/2
               C3=cos u
               C4=-cos 4u
               C5=-cos 2u
               C6=-jsin 3u
               C7=jsin u
               C8=jsin 4u
               C9=jsin 2u
```

```
T1=x(2)+x(9)
T2=x(3)+x(8)
T3=x(4)+x(7)
T4=x(5)+x(6)
T5=T1+T2+T4
T6=x(2)-x(9)
T7=x(8)-x(3)
T8=x(4)-x(7)
T9=x(5)-x(6)
T10=T6+T7+T9
T11=T1-T2
T12=T2-T4
T13=T7-T6
T14=T7-T9
T15=-T12-T11
T16=-T13+T14
M1=1*(x(1)+T3+T5)
M2=C1*T3
M3=C2*T5
M4=C3*T11
M5=C4*T12
M6=C5*T15
M7=C6*T10
M8=C6*T8
M9=C7*T13
M10=C8*T14
M11=C9*T16
S1=-M4-M5
S2=M6-M5
S3=-M9-M10
S4=M10-M11
S5=M1+M3+M3
S6=S5-M2
S7=S5+M3
S8=S6-S1
S9=S2+S6
S10=S1-S2+S6
S11=M8-S3
```

```
S12=M8-S4
S13=M8+S3+S4
X(1)=M1
X(2)=S8+S11
X(3)=S9-S12
X(4)=S7+M7
X(5)=S10+S13
X(6)=S10-S13
X(7)=S7-M7
X(8)=S9+S12
X(9)=S8-S11
```

(8) N=16; 18 complex multiplications (5), 74 complex additions, u=2π/16.

```
Coefficients:  C1=cos 2u
               C2=cos 3u
               C3=cos u+cos 3u
               C4=cos 3u-cos u
               C5=-jsin 2u
               C6=-jsin 3u
               C7=j(sin 3u-sin u)
               C8=-j(sin u+sin 3u)
```

```
T1=x(1)+x(9)
T2=x(5)+x(13)
T3=x(3)+x(11)
T4=x(3)-x(11)
T5=x(7)+x(15)
T6=x(7)-x(15)
T7=x(2)+x(10)
T8=x(2)-x(10)
T9=x(4)+x(12)
T10=x(4)-x(12)
T11=x(6)+x(14)
T12=x(6)-x(14)
T13=x(8)+x(16)
T14=x(8)-x(16)
T15=T1+T2
T16=T3+T5
T17=T15+T16
T18=T7+T11
T19=T7-T11
T20=T9+T13
T21=T9-T13
T22=T18+T20
T23=T8+T14
T24=T8-T14
T25=T10+T12
T26=T12-T10
M1=1*(T17+T22)
M2=1*(T17-T22)
M3=1*(T15-T16)
M4=1*(T1-T2)
M5=1*(x(1)-x(9))
```

```
M6=C1*(T19-T21)
M7=C1*(T4-T6)
M8=C2*(T24+T26)
M9=C3*T24
M10=C4*T26
M11=j*(T20-T18)
M12=j*(T5-T3)
M13=j*(x(13)-x(5))
M14=C5*(T19+T21)
M15=C5*(T4+T6)
M16=C6*(T23+T25)
M17=C7*T23
M18=C8*T25
S1=M4+M6
S2=M4-M6
S3=M12+M14
S4=M14-M12
S5=M5+M7
S6=M5-M7
S7=M9-M8
S8=M10-M8
S9=S5+S7
S10=S5-S7
S11=S6+S8
S12=S6-S8
S13=M13+M15
S14=M13-M15
S15=M16+M17
S16=M16-M18
S17=S13+S15
S18=S13-S15
S19=S14+S16
S20=S14-S16
X(1)=M1
X(2)=S9+S17
X(3)=S1+S3
X(4)=S12-S20
X(5)=M3+M11
X(6)=S11+S19
X(7)=S2+S4
X(8)=S10-S18
X(9)=M2
X(10)=S10+S18
X(11)=S2-S4
X(12)=S11-S19
X(13)=M3-M11
X(14)=S12+S20
X(15)=S1-S3
X(16)=S9-S17
```

Appendix B. PFA SHORT DFT ALGORITHMS

The following algorithms compute the DFT for lengths 2, 3, 4, 5, 7, 8, 9, and 16. These algorithms were taken from Burrus and Eschenbacher [11]. They are part of a complete Fortran listing of a general purpose PFA program. In contrast to Appendix A, these algorithms are written in terms of real multiplications and additions. In addition, no trivial multiplications are used in these algorithms. The real and imaginary parts of the complex input data are represented in natural order by XR(1), XR(2),..., XR(N) and XI(1), XI(2),..., XI(N), respectively. The complex output is stored in natural order in the XR(I) and XI(I) arrays. The values U1, U2,..., T1, T2,..., R1, R2,..., and S1, S2,... are all temporary values derived from input data and intermediate results. Generally, the operations must be performed in the order listed. The total number of real multiplications and additions required for each DFT is listed with each algorithm.

(1) N=2; 0 real multiplications, 4 real additions

```
T1=XR(1)
XR(1)=T1+XR(2)
XR(2)=T1-XR(2)
T1=XI(1)
XI(1)=T1+XI(2)
XI(2)=T1-XI(2)
```

(2) N=3; 4 real multiplications, 12 real additions, $u=2\pi/3$.

```
Coefficients:  C1=sin u
               C2=1/2
```

```
T1=(XR(2)-XR(3))*C1
U1=(XI(2)-XI(3))*C1
R1=XR(2)+XR(3)
S1=XI(2)+XI(3)
T2=XR(1)-R1*C2
U2=XI(1)-S1*C2
XR(1)=XR(1)+R1
XI(1)=XI(1)+S1
XR(2)=T2+U1
XR(3)=T2-U1
XI(2)=U2-T1
XI(3)=U2+T1
```

(3) N=4, 0 real multiplications, 16 real additions.

```
R1=XR(1)+XR(3)
R2=XR(1)-XR(3)
S1=XI(1)+XI(3)
S2=XI(1)-XI(3)
R3=XR(2)+XR(4)
R4=XR(2)-XR(4)
S3=XI(2)+XI(4)
S4=XI(2)-XI(4)
XR(1)=R1+R3
XR(3)=R1-R3
XI(1)=S1+S3
XI(3)=S1-S3
XR(2)=R2+S4
XR(4)=R2-S4
XI(2)=S2-R4
XI(4)=S2+R4
```

(4) N=5, 10 real multiplications, 34 real additions, u=2π/5.

```
Coefficients:  C1=sin u
               C2=sin u+sin 2u
               C3=sin u-sin 2u
               C4=(cos u-cos 2u)/2
               C5=-5/4
```

```
R1=XR(2)+XR(5)
R2=XR(2)-XR(5)
S1=XI(2)+XI(5)
S2=XI(2)-XI(5)
R3=XR(3)+XR(4)
R4=XR(3)-XR(4)
S3=XI(3)+XI(4)
S4=XI(3)-XI(4)
T1=(R2+R4)*C1
U1=(S2+S4)*C1
R2=T1-R2*C2
S2=U1-S2*C2
R4=T1-R4*C3
S4=U1-S4*C3
T1=(R1-R3)*C4
U1=(S1-S3)*C4
T2=R1+R3
U2=S1+S3
XR(1)=XR(1)+T2
XI(1)=XI(1)+U2
T2=XR(1)+T2*C5
U2=XI(1)+U2*C5
R1=T2+T1
R3=T2-T1
S1=U2+U1
S3=U2-U1
XR(2)=R1+S4
XR(5)=R1-S4
XI(2)=S1-R4
XI(5)=S1+R4
XR(3)=R3-S2
XR(4)=R3+S2
XI(3)=S3+R2
XI(4)=S3-R2
```

(5) N=7, 16 real multiplications, 72 real additions, u=2π/7.

```
Coefficients:  C1=-7/6
```

```
                    C2=(2cos u-cos 2u-cos 3u)/3
                    C3=(cos u-2cos 2u+cos 3u)/3
                    C4=(cos u+cos 2u-2cos 3u)/3
                    C5=(sin u+sin 2u-sin 3u)/3
                    C6=(2sin u-sin 2u+sin 3u)/3
                    C7=(-sin u+2sin 2u+sin 3u)/3
                    C8=(sin u+sin 2u+ 2sin 3u)/3

R1=XR(2)+XR(7)
R2=XR(2)-XR(7)
S1=XI(2)+XI(7)
S2=XI(2)-XI(7)
R3=XR(3)+XR(6)
R4=XR(3)-XR(6)
S3=XI(3)+XI(6)
S4=XI(3)-XI(6)
R5=XR(4)+XR(5)
R6=XR(4)-XR(5)
S5=XI(4)+XI(5)
S6=XI(4)-XI(5)
T1=R1+R3+R5
U1=S1+S3+S5
XR(1)=XR(1)+T1
XI(1)=XI(1)+U1
T1=XR(1)+C1*T1
U1=XI(1)+C1*U1
T2=C2*(R1-R5)
U2=C2*(S1-S5)
T3=C3*(R5-R3)
U3=C3*(S5-S3)
T4=C4*(R3-R1)
U4=C4*(S3-S1)
R1=T1+T2+T3
R3=T1-T2-T4
R5=T1-T3+T4
S1=U1+U2+U3
S3=U1-U2-U4
S5=U1-U3+U4
[illegible]=C5*(S2+S4-S6)
U1=C5*(R2+R4-R6)
T2=C6*(R2+R6)
U2=C6*(S2+S6)
T3=C7*(R4+R6)
U3=C7*(S4+S6)
T4=C8*(R4-R2)
U4=C8*(S4-S2)
R2=T1+T2+T3
R4=T1-T2-T4
R6=T1-T3+T4
S2=U1+U2+U3
S4=U1-U2-U4
S6=U1-U3+U4
XR(2)=R1+S2
XR(7)=R1-S2
```

```
XI(2)=S1-R2
XI(7)=S1+R2
XR(3)=R3+S4
XR(6)=R3-S4
XI(3)=S3-R4
XI(6)=S3+R4
XR(4)=R5-S6
XR(5)=R5+S6
XI(4)=S5+R6
XI(5)=S5-R6
```

(6) N=8, 4 real multiplications, 52 real additions, u=2π/8.

Coefficients: C1=sin u

```
R1=XR(1)+XR(5)
R2=XR(1)-XR(5)
S1=XI(1)+XI(5)
S2=XI(1)-XI(5)
R3=XR(2)+XR(8)
R4=XR(2)-XR(8)
S3=XI(2)+XI(8)
S4=XI(2)-XI(8)
R5=XR(3)+XR(7)
R6=XR(3)-XR(7)
S5=XI(3)+XI(7)
S6=XI(3)-XI(7)
R7=XR(4)+XR(6)
R8=XR(4)-XR(6)
S7=XI(4)+XI(6)
S8=XI(4)-XI(6)
T1=R1+R5
T2=R1-R5
U1=S1+S5
U2=S1-S5
T3=R3+R7
R3=C1*(R3-R7)
U3=S3+S7
S3=C1*(S3-S7)
T4=R4-R8
R4=C1*(R4+R8)
U4=S4-S8
S4=C1*(S4+S8)
T5=R2+R3
T6=R2-R3
U5=S2+S3
U6=S2-S3
T7=R4+R6
T8=R4-R6
U7=S4+S6
U8=S4-S6
XR(1)=T1+T3
XR(5)=T1-T3
XI(1)=U1+U3
XI(5)=U1-U3
XR(2)=T5+U7
XR(8)=T5-U7
XI(2)=U5-T7
XI(8)=U5+T7
XR(3)=T2+U4
XR(7)=T2-U4
XI(3)=U2-T4
XI(7)=U2+T4
XR(4)=T6+U8
XR(6)=T6-U8
XI(4)=U6-T8
XI(6)=U6+T8
```

(7) N=9, 20 real multiplications, 84 real additions, u=2π/9.

```
Coefficients:  C1=sin 3u
               C2=1/2
               C3=-cos 4u
               C4=-cos 2u
               C5=cos u
               C6=-sin 4u
               C7=-sin 2u
               C8=-sin u

R1=XR(2)+XR(9)
R2=XR(2)-XR(9)
S1=XI(2)+XI(9)
S2=XI(2)-XI(9)
R3=XR(3)+XR(8)
R4=XR(3)-XR(8)
S3=XI(3)+XI(8)
S4=XI(3)-XI(8)
R5=XR(4)+XR(7)
T1=C1*(XR(7)-XR(4))
S5=XI(4)+XI(7)
U1=C1*(XI(7)-XI(4))
R7=XR(5)+XR(6)
R8=XR(5)-XR(6)
S7=XI(5)+XI(6)
S8=XI(5)-XI(6)
R9=XR(1)+R5
S9=XI(1)+S5
T2=XR(1)-R5*C2
U2=XI(1)-S5*C2
T3=(R3-R7)*C3
U3=(S3-S7)*C3
T4=(R1-R7)*C4
U4=(S1-S7)*C4
T5=(R1-R3)*C5
U5=(S1-S3)*C5
R10=R1+R3+R7
S10=S1+S3+S7
R1=T2+T3+T5
```

```
R3=T2-T3-T4
R7=T2+T4-T5
S1=U2+U3+U5
S3=U2-U3-U4
S7=U2+U4-U5
XR(1)=R9+R10
XI(1)=S9+S10
R5=R9-R10*C2
S5=S9-S10*C2
R6=-(R2-R4+R8)*C1
S6=-(S2-S4+S8)*C1
T3=(R4+R8)*C6
U3=(S4+S8)*C6
T4=(R2-R8)*C7
U4=(S2-S8)*C7
T5=(R2+R4)*C8
U5=(S2+S4)*C8
R2=T1+T3+T5
R4=T1-T3-T4
R8=T1+T4-T5
S2=U1+U3+U5
S4=U1-U3-U4
S8=U1+U4-U5
XR(2)=R1-S2
XR(9)=R1+S2
XI(2)=S1+R2
XI(9)=S1-R2
XR(3)=R3+S4
XR(8)=R3-S4
XI(3)=S3-R4
XI(8)=S3+R4
XR(4)=R5-S6
XR(7)=R5+S6
XI(4)=S5+R6
XI(7)=S5-R6
XR(5)=R7-S8
XR(6)=R7+S8
XI(5)=S7+R8
XI(6)=S7-R8
```

(8) N=16, 20 real multiplications, 148 real additions, u=2π/16.

```
Coefficients:  C1=sin 2u
               C2=sin u
               C3=cos u+sin u
               C4=cos u-sin u
               C5=cos u
```

```
R1=XR(1)+XR(9)
R2=XR(1)-XR(9)
S1=XI(1)+XI(9)
S2=XI(1)-XI(9)
R3=XR(2)+XR(10)
R4=XR(2)-XR(10)
```

```
S3=XI(2)+XI(10)
S4=XI(2)-XI(10)
R5=XR(3)+XR(11)
R6=XR(3)-XR(11)
S5=XI(3)+XI(11)
S6=XI(3)-XI(11)
R7=XR(4)+XR(12)
R8=XR(4)-XR(12)
S7=XI(4)+XI(12)
S8=XI(4)-XI(12)
R9=XR(5)+XR(13)
R10=XR(5)-XR(13)
S9=XI(5)+XI(13)
S10=XI(5)-XI(13)
R11=XR(6)+XR(14)
R12=XR(6)-XR(14)
S11=XI(6)+XI(14)
S12=XI(6)-XI(14)
R13=XR(7)+XR(15)
R14=XR(7)-XR(15)
S13=XI(7)+XI(15)
S14=XI(7)-XI(15)
R15=XR(8)+XR(16)
R16=XR(8)-XR(16)
S15=XI(8)+XI(16)
S16=XI(8)-XI(16)
T1=R1+R9
T2=R1-R9
U1=S1+S9
U2=S1-S9
T3=R3+R11
T4=R3-R11
U3=S3+S11
U4=S3-S11
T5=R5+R13
T6=R5-R13
U5=S5+S13
U6=S5-S13
T7=R7+R15
T8=R7-R15
U7=S7+S15
U8=S7-S15
T9=C1*(T4+T8)
T10=C1*(T4-T8)
U9=C1*(U4+U8)
U10=C1*(U4-U8)
R1=T1+T5
R3=T1-T5
S1=U1+U5
S3=U1-U5
R5=T3+T7
R7=T3-T7
S5=U3+U7
S7=U3-U7
```

```
R9=T2+T10
R11=T2-T10
S9=U2+U10
S11=U2-U10
R13=T6+T9
R15=T6-T9
S13=U6+U9
S15=U6-U9
T1=R4+R16
T2=R4-R16
U1=S4+S16
U2=S4-S16
T3=C1*(R6+R14)
T4=C1*(R6-R14)
U3=C1*(S6+S14)
U4=C1*(S6-S14)
T5=R8+R12
T6=R8-R12
U5=S8+S12
U6=S8-S12
T7=C2*(T2-T6)
T8=C3*T2-T7
T9=C4*T6-T7
T10=R2+T4
T11=R2-T4
R2=T10+T8
R4=T10-T8
R6=T11+T9
R8=T11-T9
U7=C2*(U2-U6)
U8=C3*U2-U7
U9=C4*U6-U7
U10=S2+U4
U11=S2-U4
S2=U10+U8
S4=U10-U8
S6=U11+U9
S8=U11-U9
T7=C5*(T1+T5)
T8=T7-C4*T1
T9=T7-C3*T5
T10=R10+T3
T11=R10-T3
R10=T10+T8
R12=T10-T8
R14=T11+T9
R16=T11-T9
U7=C5*(U1+U5)
U8=U7-C4*U1
U9=U7-C3*U5
U10=S10+U3
U11=S10-U3
S10=U10+U8
S12=U10-U8
```

```
S14=U11+U9
S16=U11-U9
XR(1)=R1+R5
XR(9)=R1-R5
XI(1)=S1+S5
XI(9)=S1-S5
XR(2)=R2+S10
XR(16)=R2-S10
XI(2)=S2-R10
XI(16)=S2+R10
XR(3)=R9+S13
XR(15)=R9-S13
XI(3)=S9-R13
XI(15)=S9+R13
XR(4)=R8-S16
XR(14)=R8+S16
XI(4)=S8+R16
XI(14)=S8-R16
XR(5)=R3+S7
XR(13)=R3-S7
XI(5)=S3-R7
XI(13)=S3+R7
XR(6)=R6+S14
XR(12)=R6-S14
XI(6)=S6-R14
XI(12)=S6+R14
XR(7)=R11-S15
XR(11)=R11+S15
XI(7)=S11+R15
XI(11)=S11-R15
XR(8)=R4-S12
XR(10)=R4+S12
XI(8)=S4+R12
XI(10)=S4-R12
```

Appendix C. SWIFT SHORT DFT ALGORITHMS

The SWIFT short DFT algorithms are given for lengths 3, 5, 7, 9, and 16. The algorithms for lengths 3 and 5 are from [9], with slight modifications. In the modified versions shown here, duplicative additions are eliminated. The SWIFT algorithms for lengths 2, 4, and 8 are identical to the PFA algorithms for the same lengths and are thus omitted. All the algorithms are written in terms of real multiplications and additions. In addition, no trivial multiplications are used in these algorithms. The real and imaginary parts of the complex input data are represented in natural order by XR(1), XR(2),..., XR(N) and XI(1), XI(2),..., XI(N), respectively. The complex output is stored in natural order in the XR and XI input arrays. The values R1, R2,..., S1, S2,..., U1, U2,..., and T1, T2,... are all temporary values derived from input data and intermediate results. The total number of real multiplications and additions for each DFT is listed with each algorithm. The algorithms listed here have not been optimized with respect to minimizing the amount of temporary storage required.

(1) N=3; 4 real multiplications, 12 real additions, u=2π/3.

```
Coefficients:  C1=-3/2
               C2=sin u

R1=XR(2)+XR(3)
R2=XR(2)-XR(3)
S1=XI(2)+XI(3)
S2=XI(2)-XI(3)
XR(1)=R1+XR(1)
XI(1)=S1+XI(1)
T1=R1*C1
T2=R2*C2
U1=S2*C2
U2=S1*C1
T3=XR(1)+T1
U3=XI(1)+U2
XR(2)=T3+U1
XR(3)=T3-U1
XI(2)=U3-T2
XI(3)=U3+T2
```

(2) N=5, 16 real multiplications, 32 real additions, u=2π/5.

```
Coefficients:  C1=cos u-1
               C2=cos 2u-1
               C3=sin u
               C4=sin 2u

R1=XR(2)+XR(5)
R2=XR(2)-XR(5)
S1=XI(2)+XI(5)
S2=XI(2)-XI(5)
R3=XR(3)+XR(4)
R4=XR(3)-XR(4)
S3=XI(3)+XI(4)
```

```
S4=XI(3)-XI(4)
T1=R1+R3
U1=S1+S3
XR(1)=XR(1)+T1
XI(1)=XI(1)+U1
T2=XR(1)+(C1*R1)+(C2*R3)
T3=XR(1)+(C2*R1)+(C1*R3)
T4=(C3*R2)+(C4*R4)
T5=(C4*R2)-(C3*R4)
U2=(C3*S2)+(C4*S4)
U3=(C4*S2)-(C3*S4)
U4=XI(1)+(C1*S1)+(C2*S3)
U5=XI(1)+(C2*S1)+(C1*S3)
XR(2)=T2+U2
XR(3)=T3+U3
XR(4)=T3-U3
XR(5)=T2-U2
XI(2)=U4-T4
XI(3)=U5-T5
XI(4)=U5+T5
XI(5)=U4+T4
```

(3) N=7, 36 real multiplications, 60 real additions, u=2π/7.

```
Coefficients:  C1=cos u
               C2=cos 2u
               C3=cos 3u
               C4=sin u
               C5=sin 2u
               C6=sin 3u
```

```
R1=XR(2)+XR(7)
R2=XR(2)-XR(7)
S1=XI(2)+XI(7)
S2=XI(2)-XI(7)
R3=XR(3)+XR(6)
R4=XR(3)-XR(6)
S3=XI(3)+XI(6)
S4=XI(3)-XI(6)
R5=XR(4)+XR(5)
R6=XR(4)-XR(5)
S5=XI(4)+XI(5)
S6=XI(4)-XI(5)
T1=R1+R3+R5
U1=S1+S3+S5
XR(1)=XR(1)+T1
XI(1)=XI(1)+U1
T2=XR(1)+(C1*R1)+(C2*R3)+(C3*R5)
T3=XR(1)+(C2*R1)+(C3*R3)+(C1*R5)
T4=XR(1)+(C3*R1)+(C1*R3)+(C2*R5)
T5=(C4*R2)+(C5*R4)+(C6*R6)
T6=(C5*R2)-(C6*R4)-(C4*R6)
T7=(C6*R2)-(C4*R4)+(C5*R6)
U2=(C4*S2)+(C5*S4)+(C6*S6)
```

```
U3=(C5*S2)-(C6*S4)-(C4*S6)
U4=(C6*S2)-(C4*S4)+(C5*S6)
U5=XI(1)+(C1*S1)+(C2*S3)+(C3*S5)
U6=XI(1)+(C2*S1)+(C3*S3)+(C1*S5)
U7=XI(1)+(C3*S1)+(C1*S3)+(C2*S5)
XR(2)=T2+U2
XR(3)=T3+U3
XR(4)=T4+U4
XR(5)=T4-U4
XR(6)=T3-U3
XR(7)=T2-U2
XI(2)=U5-T5
XI(3)=U6-T6
XI(4)=U7-T7
XI(5)=U5+T5
XI(6)=U6+T6
XI(7)=U7+T7
```

(4) N=9, 44 real multiplications, 88 real additions, u=2π/9.

```
Coefficients:  C1=cos u
               C2=cos 2u
               C3=cos 3u
               C4=cos 4u
               C5=sin u
               C6=sin 2u
               C7=sin 3u
               C8=sin 4u
```

```
R1=XR(2)+XR(9)
R2=XR(2)-XR(9)
S1=XI(2)+XI(9)
S2=XI(2)-XI(9)
R3=XR(3)+XR(8)
R4=XR(3)-XR(8)
S3=XI(3)+XI(8)
S4=XI(3)-XI(8)
R5=XR(4)+XR(7)
R6=XR(4)-XR(7)
S5=XI(4)+XI(7)
S6=XI(4)-XI(7)
R7=XR(5)+XR(6)
R8=XR(5)-XR(6)
S7=XI(5)+XI(6)
S8=XI(5)-XI(6)
T1=R1+R3+R5+R7
U1=S1+S3+S5+S7
XR(1)=XR(1)+T1
XI(1)=XI(1)+U1
T2=(C3*R5)+XR(1)
T3=(C1*R1)+(C2*R3)+(C4*R7)+T2
T4=(C2*R1)+(C4*R3)+(C1*R7)+T2
T5=C3*(T1-R5)+R5+XR(1)
T6=(C4*R1)+(C1*R3)+(C2*R7)+T2
```

```
T7=C7*R6
T8=(C5*R2)+(C6*R4)+(C8*R8)+T7
T9=(C6*R2)+(C8*R4)-(C5*R8)-T7
T10=C7*(R2-R4+R8)
T11=(C8*R2)-(C5*R4)-(C6*R8)+T7
U2=C7*S6
U3=(C5*S2)+(C6*S4)+(C8*S8)+U2
U4=(C6*S2)+(C8*S4)-(C5*S8)-U2
U5=C7*(S2-S4+S8)
U6=(C8*S2)-(C5*S4)-(C6*S8)+U2
U7=(C3*S5)+XI(1)
U8=(C1*S1)+(C2*S3)+(C4*S7)+U7
U9=(C2*S1)+(C4*S3)+(C1*S7)+U7
U10=C3*(U1-S5)+S5+XI(1)
U11=(C4*S1)+(C1*S3)+(C2*S7)+U7
XR(2)=T3+U3
XR(3)=T4+U4
XR(4)=T5+U5
XR(5)=T6+U6
XR(6)=T6-U6
XR(7)=T5-U5
XR(8)=T4-U4
XR(9)=T3-U3
XI(2)=U8-T8
XI(3)=U9-T9
XI(4)=U10-T10
XI(5)=U11-T11
XI(6)=U11+T11
XI(7)=U10+T10
XI(8)=U9+T9
XI(9)=U8+T8
```

(5) N=16, 24 real multiplications, 144 real additions, u=2π/16.

Coefficients: C1=cos u
C2=cos 2u
C3=cos 3u

```
R1=XR(1)+XR(9)
R2=XR(1)-XR(9)
S1=XI(1)+XI(9)
S2=XI(1)-XI(9)
R3=XR(2)+XR(16)
R4=XR(2)-XR(16)
S3=XI(2)+XI(16)
S4=XI(2)-XI(16)
R5=XR(3)+XR(15)
R6=XR(3)-XR(15)
S5=XI(3)+XI(15)
S6=XI(3)-XI(15)
R7=XR(4)+XR(14)
R8=XR(4)-XR(14)
S7=XI(4)+XI(14)
S8=XI(4)-XI(14)
```

```
R9=XR(5)+XR(13)
R10=XR(5)-XR(13)
S9=XI(5)+XI(13)
S10=XI(5)-XI(13)
R11=XR(6)+XR(12)
R12=XR(6)-XR(12)
S11=XI(6)+XI(12)
S12=XI(6)-XI(12)
R13=XR(7)+XR(11)
R14=XR(7)-XR(11)
S13=XI(7)+XI(11)
S14=XI(7)-XI(11)
R15=XR(8)+XR(10)
R16=XR(8)-XR(10)
S15=XI(8)+XI(10)
S16=XI(8)-XI(10)
T1=R13+R5
T2=R13-R5
T3=R1+R9
T4=R1-R9
T5=T3+T1
T6=T3-T1
T7=C2*T2
T8=R2-T7
T9=R2+T7
T10=R3+R15
T11=R3-R15
T12=R7+R11
T13=R7-R11
T14=T10+T12
T15=(C1*T11)+(C3*T13)
T16=C2*(T10-T12)
T17=(C3*T11)-(C1*T13)
T18=C2*(R6+R14)
T19=R6-R14
T20=T18+R10
T21=T18-R10
T22=R4+R16
T23=R4-R16
T24=R8+R12
T25=R8-R12
T26=(C3*T22)+(C1*T24)
T27=C2*(T23+T25)
T28=(C1*T22)-(C3*T24)
T29=T23-T25
U1=S1+S9
U2=S1-S9
U3=S5+S13
U4=S5-S13
U5=U1+U3
U6=C2*U4
U7=S2+U6
U8=S2-U6
U9=U1-U3
```

```
U10=S3+S15
U11=S3-S15
U12=S11+S7
U13=S11-S7
U14=U10+U12
U15=C2*(U10-U12)
U16=(C1*U11)-(C3*U13)
U17=(C3*U11)+(C1*U13)
U18=C2*(S6+S14)
U19=U18+S10
U20=U18-S10
U21=S6-S14
U22=S4+S16
U23=S4-S16
U24=S8+S12
U25=S8-S12
U26=(C3*U22)+(C1*U24)
U27=C2*(U23+U25)
U28=(C1*U22)-(C3*U24)
U29=U23-U25
XR(1)=T5+T14
XI(1)=U5+U14
XR(5)=T6+U29
XI(5)=U9-T29
XR(9)=T5-T14
XI(9)=U5-U14
XR(13)=T6-U29
XI(13)=U9+T29
T30=T8+T15
U30=U26+U19
T31=T8-T15
U31=U26-U19
XR(2)=T30+U30
XR(8)=T31+U31
XR(16)=T30-U30
XR(10)=T31-U31
T32=T4+T16
U32=U27+U21
T33=T4-T16
U33=U27-U21
XR(3)=T32+U32
XR(7)=T33+U33
XR(15)=T32-U32
XR(11)=T33-U33
T34=T9+T17
U34=U28+U20
T35=T9-T17
U35=U28-U20
XR(4)=T34+U34
XR(6)=T35+U35
XR(14)=T34-U34
XR(12)=T35-U35
U36=U7+U16
T36=T20+T26
```

```
U37=U7-U16
T37=T20-T26
XI(2)=U36-T36
XI(8)=U37+T37
XI(16)=U36+T36
XI(10)=U37-T37
U38=U2+U15
T38=T19+T27
U39=U2-U15
T39=T19-T27
XI(3)=U38-T38
XI(7)=U39+T39
XI(15)=U38+T38
XI(11)=U39-T39
U40=U8+U17
T40=T21+T28
U41=U8-U17
T41=T21-T28
XI(4)=U40-T40
XI(6)=U41+T41
XI(14)=U40+T40
XI(12)=U41-T41
```

REFERENCES

1. Cooley, J. W. and Tukey, J. W., "An Algorithm for the Machine Calculation of Complex Fourier Series," Mathematics of Computation, Vol. 19, April 1965, pp. 297-301.

2. Spaulding, William G., Array Signal Processing for Radar (An Introduction and Assessment for MIRADCOM Interests), US Army Missile Command, Redstone Arsenal, Alabama, February 1979, Internal Technical Note T-79-11.

3. Good, I. J., "The Relationship Between Two Fast Fourier Transforms," IEEE Trans. Comput., Vol. C-20, March 1971, pp. 310-317.

4. Gold, B. and Rader, C. M., Digital Processing of Signals, McGraw-Hill Book Company, New York, 1969.

5. Burrus, C.S., "Index Mappings for Multidimensional Formulation of the DFT and Convolution," IEEE Trans. Acoustics, Speech, and Signal Processing, Vol. ASSP-25, June 1977, pp. 239-242.

6. Brigham, E. O., The Fast Fourier Transform, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1974.

7. Singleton, R. C., "An Algorithm for Computing the Mixed Radix Fast Fourier Transform," IEEE Trans. on Audio and Electroacoustics, Vol. AU-17, June 1969, pp. 93-103.

8. Blanken, J. D. and Rustan, P. L., "Selection Criteria for Efficient Implementation of FFT Algorithms," IEEE Trans. Acoustics, Speech, and Signal Processing, Vol. ASSP-30, February 1982, pp. 107-109.

9. Smith, W. W., Jr., Robinson, D. J., and Ward, J. R., Advanced Electronics and Digital Sensor Technology - Final Report, Vol. III - Signal Conditioning, Martin Marietta Corp. Contract No. DASG60-75-C-0060, Mod P00010 for US Army Ballistic Missile Defense Advanced Technology Center, Huntsville, AL, February 1980.

10. Kolba, D. P. and Parks, T. W., "A Prime Factor FFT Algorithm Using High-Speed Convolution," IEEE Trans, Acoustics, Speech, and Signal Processing, Vol. ASSP-25, August 1977, pp. 281-294.

11. Burrus, C. S. and Eschenbacher, P. W., "An In-Place, In-Order Prime Factor FFT Algorithm," IEEE Trans. Acoustics, Speech, and Signal Processing, Vol. ASSP-29, August 1981, pp. 806-816.

12. Winograd, S., "On Computing the Discrete Fourier Transform," Mathematics of Computation, Vol. 32, January 1978, pp. 175-199.

13. Silverman, H. F., "An Introduction to Programming the Winograd Fourier Transform Algorithm (WFTA), IEEE Trans. Acoustics, Speech and Signal Processing, Vol. ASSP-25, April 1977, pp. 152-165.

14. Silverman, H. F., "Corrections and an Addendum to "An Introduction to Programming the Winograd Fourier Transform Algorithm (WFTA), IEEE Trans, Acoustics, Speech, and Signal Processing, Vol. ASSP-26, June 1978, p. 268.

15. Tseng, B. D. and Miller, W. C., "Comments on "An Introduction to Programming the Winograd Fourier Transform Algorithm (WFTA)," IEEE Trans, Acoustics, Speech, and Signal Processing, Vol. ASSP-26, June 1978, pp. 268-269.

16. Silverman, H. F., "Further Corrections to "An Introduction to Programming the Winograd Fourier Transform Algorithm (WFTA)," IEEE Trans, Acoustics Speech, and Signal Processing, Vol. ASSP-26, October 1978, p. 482.

17. McClellan, J. H. and Nawab, H., "Complex General-N Winograd Fourier Transform Algorithm (WFTA)," in Programs for Digital Signal Processing, IEEE Press and Wiley, New York, 1979.

18. Harris, D. B., McClellan, J. H., Chan, D. S. K., and Schuessler, H. W., "Vector Radix Fast Fourier Transform," IEEE International Conference on Acoustics, Speech, and Signal Processing Record, 1977, pp. 548-551.

19. Mersereau, R. M., "The Processing of Hexagonally Sampled Two-Dimensional Signals, Proc. IEEE, Vol. 67, June 1979, pp. 930-949.

20. Sumney, L. W., "VHSIC Status and Issues," Military Electronics/Countermeasures, August 1981, pp. 77-81.

21. Martin, Jim, "Very High Speed Integrated Circuits-Into the Second Generation," Military Electronics/Countermeasures, February 1982, p. 84.

22. Aviation Week and Space Technology, "Companies Offer Different Approaches," February 16, 1981, pp. 71-72.

23. Nussbaumer, H. J., Fast Fourier Transform and Convolution Algorithms, Springer-Verlag, Berlin, 1981.

DISTRIBUTION

	No. of Copies
US Army Materiel Systems Analysis Activity	
ATTN: DRXSY-MP	1
Aberdeen Proving Ground, Maryland 21005	
Commander	
US Army Materiel Development	
and Readiness Command	
ATTN: DRCCG	1
DRCRD	1
5001 Eisenhower Avenue	
Alexandria, Virginia 22304	
Commander	
Ballistic Missile Defense Systems Command	
ATTN: BMDSC-HR	1
P.O. Box 1500	
Huntsville, Alabama 35807	
Director	
Ballistic Missile Defense	
Advanced Technology Center	
ATTN: ATC-R	2
P.O. Box 1500	
Huntsville, Alabama 35807	
Commander	
US Army Electronics Research	
and Development Command	
ATTN: DRSEL, Mr. Fishbien	1
DRCPM-MALR	1
Fort Monmouth, New Jersey	
IIT Research Institute	
ATTN: GACIAC	1
10 West 35th Street	
Chicago, Illinois 60616	
DRCPM-MD-T-C, Mr. Evans	1
-HAE, Mr. Ams	1
-ROL, Mr. Bishop	1
DRSMI-LP, Mr. Voigt	1

DISTRIBUTION (Concluded)

		No. of Copies
DRSMI-R,	Dr. McCorkle	1
-R,	Dr. Rhoades	1
-RE,	Mr. Lindberg	1
-RE,	Mr. Pittman	1
-REI,	Mr. Currie	1
-REG,	Mr. Root	1
-RER,	Mr. Low	1
-RER,	Mr. Hicks	50
-RE,	(Record Set)	1
-RG,	Mr. Yates	1
-RPR		15
-RPT	(Record Set)	1